Deploying Affect-Inspired Mechanisms to Enhance Agent Decision-Making and Communication

Thesis advisor

Barbara J. Grosz

Author

Dimitrios Antos

Deploying Affect-Inspired Mechanisms to Enhance Agent Decision-Making and Communication

Abstract

Computer agents are required to make appropriate decisions quickly and efficiently. As the environments in which they act become increasingly complex, efficient decision-making becomes significantly more challenging. This thesis examines the positive ways in which human emotions influence people's ability to make good decisions in complex, uncertain contexts, and develops computational analogues of these beneficial functions, demonstrating their usefulness in agent decision-making and communication.

For decision-making by a single agent in large-scale environments with stochasticity and high uncertainty, the thesis presents GRUE (Goal Re-prioritization Using Emotion), a decision-making technique that deploys emotion-inspired computational operators to dynamically re-prioritize the agent's goals. In two complex domains, GRUE is shown to result in improved agent performance over many existing techniques.

Agents working in groups benefit from communicating and sharing information that would otherwise be unobservable. The thesis defines an affective signaling mechanism, inspired by the beneficial communicative functions of human emotion, that increases coordination. In two studies, agents using the mechanism are shown to

make faster and more accurate inferences than agents that do not signal, resulting in improved performance. Moreover, affective signals confer performance increases equivalent to those achieved by broadcasting agents' entire private state information.

Emotions are also useful signals in agents' interactions with people, influencing people's perceptions of them. A computer-human negotiation study is presented, in which virtual agents expressed emotion. Agents whose emotion expressions matched their negotiation strategy were perceived as more trustworthy, and they were more likely to be selected for future interactions.

In addition, to address similar limitations in strategic environments, this thesis uses the theory of reasoning patters in complex game-theoretic settings. An algorithm is presented that speeds up equilibrium computation in certain classes of games. For Bayesian games, with and without a common prior, the thesis also discusses a novel graphical formalism that allows agents' possibly inconsistent beliefs to be succinctly represented, and for reasoning patterns to be defined in such games. Finally, the thesis presents a technique for generating advice from a game's reasoning patterns for human decision-makers, and demonstrates empirically that such advice helps people make better decisions in a complex game.

Contents

Citations to Previously Published Work

Large portions of Chapters 2 – 6 have appeared in the following papers:

"The influence of emotion expression on perceptions of trustworthiness in negotiation", Dimitrios Antos, Celso De Melo, Jonathan Gratch and Barbara Grosz, In the Proceedings of the Twenty-Fifth Conference on Artificial Intelligence (AAAI), San Francisco, CA, August 2011;

"Using Emotions to Enhance Decision-Making", Dimitrios Antos and Avi Pfeffer, In the Proceedings of the Twenty-Second Joint Conference on Artificial Intelligence (IJCAI), Barcelona, Catalonia, Spain, July 2011;

"Reasoning Patterns in Bayesian Games", *extended abstract*, Dimitrios Antos and Avi Pfeffer, In the Proceedings of the Conference on Autonomous Agents and Multi-Agent Systems (AAMAS), Taipei, Taiwan, May 2011;

"Representing Bayesian Games with Non-Common Priors", *extended abstract*, Dimitrios Antos and Avi Pfeffer, In the Proceedings of the Conference on Autonomous Agents and Multi-Agent Systems (AAMAS), Toronto, Ontario, Canada, May 2010;

"Using Reasoning Patterns to Help Humans Solve Complex Games", Dimitrios Antos and Avi Pfeffer, In the Proceedings of the Twenty-First Joint Conference on Artificial Intelligence (IJCAI), Pasadena, CA, July 2009;

"Simplifying Games Using Reasoning Patterns", *student abstract*, Dimitrios Antos and Avi Pfeffer, In the Proceedings of the Twenty-Second Conference on Artificial Intelligence (AAAI), Chicago, IL, July 2008;

"Identifying Reasoning Patterns in Games", Dimitrios Antos and Avi Pfeffer, In the Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI), Helsinki, Finland, June 2008.

Acknowledgments

This thesis would not have existed, were it not for the support, encouragement, and guidance of a large number of people. A great many thanks are owed to my advisor, Professor Barbara Grosz, and also to my former advisor, Professor Avi Pfeffer, for their continual mentorship and assistance, academically, financially and personally. My gratitude is also extended to Professor David Parkes for giving me invaluable advice throughout my graduate studies, and for granting me the opportunity to practice and sharpen my teaching skills and interact with a large number of incredibly talented students in the process. Professor Krzysztof Gajos, also, has granted me the privilege to be part of his research group, and has consistently been an approachable and most useful mentor, for which I am most grateful.

Special thanks goes to Professor Jonathan Gratch, who twice admitted to the summer internship program of the Institute for Creative Technologies (ICT) of the University of Southern California. These two summers were not only immensely enjoyable, but his guidance helped give form and shape to my ideas, making this thesis possible. Along with him, I must also thank two more professors of ICT, Louis-Philippe (LP) Morency and Stacy Marsella, as well as a large number of other colleagues and interns there, above all Celso De Melo for being a consistent and most passionate collaborator and coauthor.

Several courses at Harvard shaped my thoughts and equipped me with the knowledge and skills to tackle interesting research problems. I would like to especially thank Harvard Professors Stuart Shieber and David Laibson for teaching me about Computational Linguistics and Behavioral Economics, respectively, as well as MIT Professor Rosalind Picard for allowing me to attend her amazing Affective Computing

course.

A doctoral thesis is not possible without the support of individuals that dilute graduate school's stressful times and enrich the good ones. My colleagues at the Harvard *Artificial Intelligence Research Group* (AIRG), the Harvard *Mind, Brain and Behavior* (MBB) society, and especially my long-term office mates and friends, Kobi Gal, Ece Kamar, Philip Hendrix, Katharina Reinecke, Lahiru Jayatilaka, Anna Huang, and Elena Agapie, deserve special thanks for making grad school fun. My life maintained a healthy balance thanks to my roommates, especially Greg Hyde, and my partner Travis Emick. What I am, however, I owe almost exclusively to my parents, Antonios and Foteini, and to my sister Nikoleta—I could not imagine ever doing anything that would repay their love, support, encouragement, and sacrifices.

Last but not least, I would like to acknowledge my funding sources, in particular the *National Science Foundation* (NSF grant #0705406), the *Air Force Office of Sponsored Research* (AFOSR) under the Multi-University Research Initiative (MURI) contract FA9550-05-1- 032 (award number 5710001863), the Harvard *Mind, Brain and Behavior* society for twice granting me their graduate student award, the Institute for Creative Technologies for their generous support during my internships there, and the *Siebel Foundation* for awarding me their scholarship in my final year of graduate studies.

To my parents, whose support made this all possible, and to my partner Travis, for making my life during those years a truly happy one.

Chapter 1

Introduction

Computer agents are required to make appropriate decisions in a large variety of situations. By themselves, collaborating with other agents, or interacting with people, agents must evaluate their options and take appropriate actions, and do so quickly and efficiently. Decision-making has become increasingly challenging for computer agents, however, as the environments and situations in which they act are becoming more and more complex. This complexity manifests in terms of the domain's increasing size, the world's stochasticity, and the level of uncertainty the agents face. Research in psychology has demonstrated that people's emotions enable them to make good decisions in such situations. Drawing upon prior work that has provided high-fidelity computational models of human emotions, this thesis builds affect-inspired mechanisms to address the difficult and challenging problem of decision-making by computer agents in complex domains.

To illustrate the computational challenges of complexity, we shall contrast two different decision-making situations for agents: In the first, an agent plays a game

of tic-tac-toe against a human. In the second, an agent trades stocks in a financial market. Tic-tac-toe consists of a small rectangular (3×3) board and has very simple rules. Moreover, an agent designed to play tic-tac-toe may be uncertain about the future actions of its opponent, but it has no uncertainty about the rules of the game or either player's objectives. On the other hand, financial markets present a very complex situation. The agent must continually monitor the stochastic price movements of hundreds of stocks. To achieve superior performance, it may also need to be aware of new developments in the economy and anticipate their effect on prices. Furthermore, the actions of other traders in the market are typically not observable, increasing the agent's uncertainty.

A number of approaches in AI have been used to deal with the challenge of complexity. Agents have been designed that operate successfully by exploiting the world's structure. To tackle uncertainty, machine learning has been deployed to make the environment more predictable. Richer new models have been developed that allow agents to represent the world and reason about it in more nuanced and efficient ways.

This thesis takes a different approach. It looks to the positive ways in which emotion influences people's decision-making and builds computational analogues of these beneficial functions for agents to make decisions. As demonstrated by psychologists, emotion influences decision-making in a number of beneficial ways [55, 85, 42, 26, 73, 11, 96, 40]. To illustrate some of these, consider this scenario: You are walking through a beautiful forest in the springtime, enjoying the sight of flowers and the refractions of the sun's rays, when you suddenly notice a big angry bear rushing towards you from at a distance. In this situation, most people would experience the emotion

of fear, which carries helpful cognitive, communicative and physiological effects. Cognitively, emotions help narrow perceptual attention to what is very important in the situation at hand; in the case of fear, attention is focused on self-preservation cues, such as locating escape routes. Emotion may also lead to more myopic, short-term decision-making. Although in some cases this can be harmful to decision-making, it may also simplify a person's reasoning under time pressure. Affect also has a communicative function; a person who becomes afraid typically broadcasts this internal state by means of facial expressions, prosody and body movements, which can alert others and therefore increase the survival chances of a group of people.

Prior work has constructed a number of computational models of emotion [50, 133, 118]. These computational models rely on theories of emotion from psychology, most notably *cognitive appraisal* theories [73, 41] which describe emotion as the result of an *appraisal* of a situation with respect to the person's goals and desires. These models have been designed with the aim of replicating emotion manifestation in people in a high-fidelity manner, and have been used to test the predictions of theories, or build virtual agents that behave in human-like ways.

This thesis builds novel computational analogues of human emotion. It does not aim, however, to replicate the way emotions operate in people, but to assist agents in their decision-making. In this way, the thesis addresses the big challenge of identifying those helpful functions of affect and successfully incorporating them into the design of computer agents. In a number of different settings, the thesis demonstrates that these computational analogues of emotion can improve agents' decision-making performance in complex domains, and can facilitate better communication between

them and other agents, as well as between them and people.

In particular, for a single agent making decisions in a complex environment, a set of emotion-inspired computational operators are designed. These operators are used to dynamically re-prioritize the agent's goals. In every point in time, the agent chooses actions myopically toward achieving its higher-priority goals. This mechanism helps in two ways: On the one hand, it addresses the problem of large scale: the fact that the agent optimizes myopically helps avoid considering a very large number of possible future contingencies. On the other hand, it assists with uncertainty and stochasticity: instead of explicitly tracking the unknowns and the changing state of the world, it relies on the emotion-inspired computational operators to adjust its priorities, and thus adapt its behavior to changing circumstances. The usefulness of the mechanism is demonstrated in two complex domains. The first is the *restless bandits* problem, which has been shown to be intractable in the general case [101] and approximations have only been designed for very specific cases, in which the stochasticity of the world is known to the agent [54], but not for cases in which the agent is uncertain about how the world changes. A foraging task domain is also used in order to illustrate the usefulness of the mechanism in a world with several millions of states. In both domains the emotion-inspired mechanism outperformed other techniques, including a reactive algorithm and a learning agent.

For groups of agents collaborating or competing in an environment, an effective affective signaling mechanism has been designed. In particular, the mechanism focuses on the problem of agents possessing privately known characteristics that are unobservable by others. The presence of such characteristics increases the uncertainty of

agents interacting with each other, and it can impede successful coordination among them. The mechanism continually computes a set of appraisal variables that reflect the agent's attitude toward what is happening. These variables are used to then generate and broadcast a signal that is analogous to human emotions like joy or sadness. Agents receiving the signal perform Bayesian updating to infer the unobservable characteristics of the sender. The usefulness of the mechanism is illustrated in a simple repeated social dilemma, and then in a much more complex task domain. In both cases, agents that implement the mechanism make faster and more accurate inferences than agent who exchange no signals, which also leads to improved performance and enhanced coordination. Interestingly, affective signaling in the second domain leads to performance increases that are not significantly different from those of communicating agents' entire private state information. This result suggests that affective signals efficiently convey information that is most relevant to decision-making.

Affective signals are also shown to be useful in computers' interaction with people. In particular, they can influence human perceptions of agents' traits. A study is presented that focuses on one of those traits, the agents' trustworthiness. People were asked to negotiate with a number of computer agents in order to split a set of valuable resources. Agents differed both in terms of their negotiation strategy, as well as in terms of whether they displayed emotion expressions on a virtual agent face. After a playing a number of games with various agents that differed in their emotion expressions, but not in their negotiation strategy, each subject was asked to select one of them to play a "trust game." People's selection was used as a measure of how trustworthy they perceived each agent to be. They preferred agents who

expressed emotion over those who did not. More importantly, they showed significant preference for agents whose emotion expressions matched their negotiation strategy. For instance, among tough negotiators, those who expressed negative emotion (anger) were seen as more trustworthy; among flexible and conciliatory negotiators, those who smiled were preferred.

Strategic environments are another place where cognitive and computational limitations are important, and for such environments it is typical to adopt a game-theoretic analysis. The second part of my thesis considers the role of reasoning patterns [103], or qualitative styles of reasoning, in more efficiently representing, reasoning about, and solving complex games, as well as in assisting humans make better decisions in complex environments.

The theory of reasoning patterns analyzes games by describing in succinct ways the motivation of agents. In particular, it tracks the flow of utility and information within a game, which enables a modeler of the game to describe why agents behave the way they do, or to identify good strategies for them. I use the theory of reasoning patterns in three ways:

First, I introduce a new graphical formalism for Bayesian games. This formalism allows agents' beliefs to be represented succinctly in a "belief graph," even if they are mutually inconsistent. It also helps extend the definition of reasoning patterns to a particularly complex type of Bayesian games, those without a common prior. This extension makes it possible to use the reasoning patterns in such games to understand agents' behavior and identify well-performing strategies for them.

Second, I use the reasoning patterns to simplify games for the purpose of comput-

ing a Nash equilibrium of them. An algorithm is described that identifies a game's reasoning pattern in polynomial time. The algorithm discovers decisions within the game which can be ignored in computing an equilibrium. In certain classes of games, this results in exponential time savings.

Finally, I use the theory to assist people in complex environments. An agent is built that generates advice for people making decisions. This advice offers insights regarding the possible effects of their actions, which it does by examining the game's reasoning patterns. The usefulness of this advice is tested empirically by asking people to play a complex repeated game of incomplete information. Subjects who received the advice outperformed subjects who did not have access to it.

1.1 Emotions

Throughout history, emotions have been the topic of much debate in philosophy and psychology [121]. Among the ancients, Plato and Aristotle take the position that emotions are an integral part of the mind (or "soul") and can be deployed either wisely or unwisely. For instance, when anger is directed towards punishing the unjust it is wisely deployed, but not so when it manifests in the desire to commit crimes. Others, such as the stoic philosophers, advocate a silencing of the "passions" in favor of *apatheia*, a calm and detached stance towards the changeability of the world. In later centuries, Christian philosophers distinguished between virtuous emotions, such as compassion, and sinful ones, like anger. Modern writers expressed a variety of opinions on the emotions, some treating them as enemies of reason (Descartes), others as assistants of rational and moral thinking (Kant) and some as superior to it

(Nietzche). Overall, emotions were seen as secondary to reason and people's attention was captivated largely by instances in which they carry people astray. It was only during the 20th century that emotions became officially a topic of rigorous scientific study, and their role was better understood.

Our modern understanding of the emotions comes from cognitive psychology and neuroscience. We now know that emotion exhibits widespread manifestation in the brain and is not limited to a particular "emotion center." And although psychologists still disagree on the specifics of what emotion is, all acknowledge it as a complex cognitive and physiological phenomenon with extensive influences on perception, attention, reasoning, decision-making, communication, and internal biochemical regulation [11, 41, 40, 55].

Emotion is typically defined in either of two ways: In *dimensional* models, emotion is mapped on a continuous, 2- or 3-dimensional space. The two most commonly used axes are *valence*, which denotes whether the person experiences a situation as positive or negative, and *arousal*, which denotes how strong his or her feelings are. For instance, both contentment and delight are firmly on the positive side of valence, but the latter is more intense than the former. (See Figure 1.1 for a mapping of some emotional states along these two axes.) A third dimension with the name of *power* or *control* is also often added in those models. Fear and anger, for instance, can both be intense negative emotions, but they differ in this third dimension of control: When angry, people are empowered and driven towards taking action, while fear promotes fleeing or hiding.

In *categorical* models, emotions are not placed on a continuous space but are

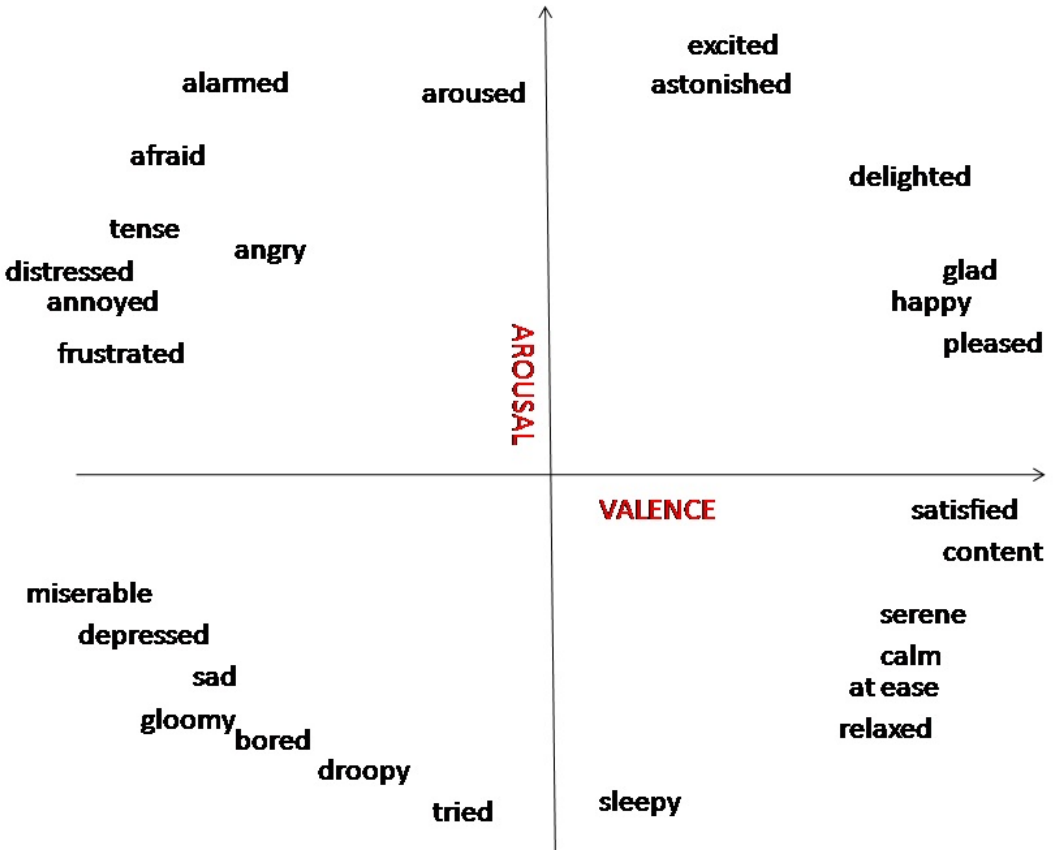

Figure 1.1: A two-dimensional mapping of some emotions

treated as discrete, separate events. Proponents of this categorical approach argue that placing different emotions closely in a continuous space cultivates misperceptions of similarity between them. They instead treat the large variety of emotions as independent phenomena. Influential in this space has been the work of Ekman [36], who identified five "basic" emotions that are expressed and recognized by all cultures (happiness, sadness, anger, fear, and disgust). Other, secondary emotions, like embarrassment, have been argued to be culture-specific. For both kinds of emotions, however, their expression and interpretation are largely modulated by cultural norms

and expectations.

Most researchers use both definitions of emotion interchangeably. A popular analogue is weather patterns: storms and fair weather can both be characterized by specific measurements on luminosity, humidity and windiness, but they can also be treated as discrete, separate phenomena. In this thesis I follow the categorical approach, without however making any express commitment to the superiority of any particular emotion definition.

The work presented in this thesis builds upon cognitive appraisal theories of emotion [6, 100, 73, 99]. These theories describe emotion as the result of an assessment (appraisal) of how a particular stimulus or observation impacts a person's goals. For instance, fear is seen as the result of appraising something as threatening to an important goal. Sadness, similarly, occurs when a situation is appraised as disastrous to a person's goals, and the person feels incapable of acting to change this.

Appraisal-based computational theories of emotion break down the appraisal process along a number of dimensions. Each dimension captures something about the way in which the observed event impacts the person's goals. Common dimensions include: *desirability*, which denotes whether the event is being appraised as better or worse than the person's expectations; *unexpectedness*, which reflects how surprising the event was to the person; *praiseworthiness* and *blameworthiness*, which denote whether the agent can attribute the event to the willful actions of another; *controllability*, which captures whether the agent can undo the effects of the observed event, and others.

To illustrate these concepts, consider the theory suggested by Ortony, Clore and

Collins (OCC, [100]), whose appraisal-based structure is shown in Figure 1.2. The OCC model was the first theory of emotion that was designed with the expressed purpose of turning it into an AI architecture. A person in the OCC model assesses a stimulus along three dimensions: The first dimension is concerned with the the desirability of the expected consequences of an event (left-side branch in Fig. 1.2). This assessment results in an undifferentiated mood (i.e., the person is pleased or displeased). Further deliberation along this line, however, may make this mood more specific. For instance, if the person assesses that the event is undesirable for another, but desirable for herself, the ensuing emotion is not just one of being pleased, but one of gloating. The second dimension (middle branch in Fig. 1.2) has to do with the role of other agents, and is a measure of their praiseworthiness or blameworthiness. If, for example, another agent (or the self) has done something worthy of praise, the ensuring emotion is one of admiration (or pride, respectively). Finally, the third dimension (right branch in Fig. 1.2) concerns long-running attributes associated with objects or people. Hence, for instance, one may feel admiration toward a colleague for one of his accomplishments (along the praiseworthiness axis), but also generally dislike him (along the axis of long-running attributes).

It is easy to see how the OCC model can be translated into an AI architecture. In fact, Ortony, Clore and Collins suggest one, in which a person's goals are hierarchically organized, such each goal may break down into sub-goals, and some goals are inhibitory towards other goals. For instance, the goal to "get a Ph.D." might consist of subgoals like "be admitted to graduate school," "do well in courses," and "write a dissertation," but might be inhibitory toward the goal "accumulate work experience

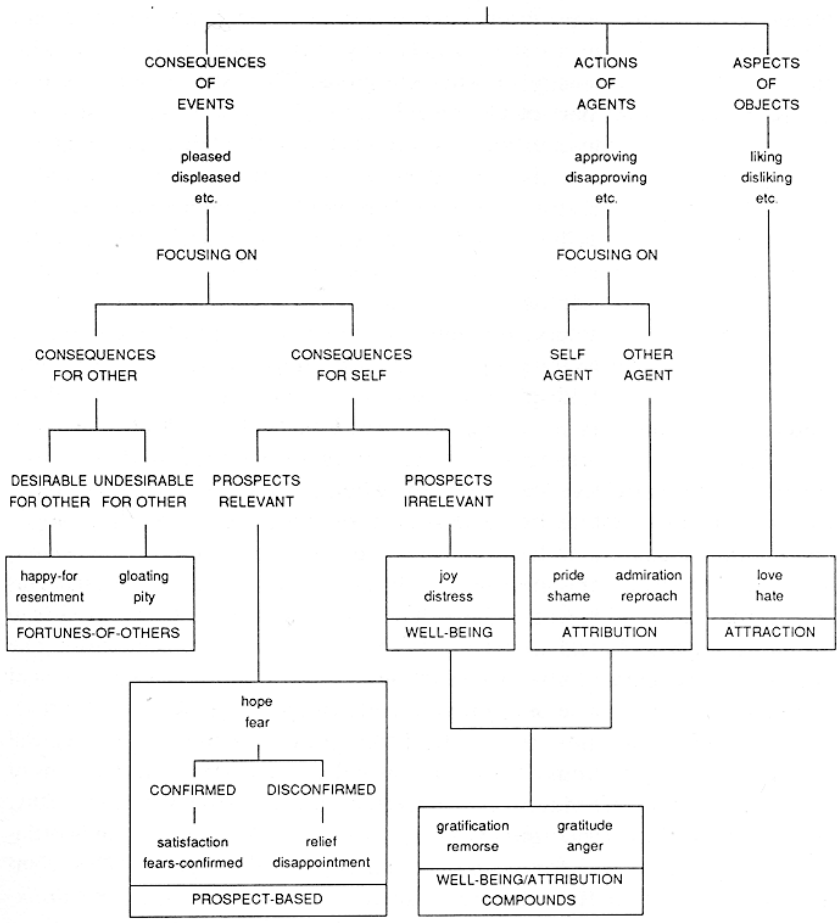

Figure 1.2: The OCC appraisal pattern

in industry early." As long as the agent maintains such a goal structure and is able to assess the impact of various observations on its goals, it can compute quantities describing the desirability of an event or the praiseworthiness (or blameworthiness) of others' actions, and maintain long-term dispositions towards people or objects (liking or disliking), and therefore produce the appropriate emotions.

This appraisal process also helps explain the effects brought about by an emotion once it emerges. Each emotion includes behavioral drives that aim to address the

situation that generated it. For example, fear emerges after a situation is appraised as threatening, and thus leads to behaviors that protect the person from the source of the threat, like hiding, fleeing, or defending himself. Sadness, similarly, emerges when the person feels unable to deal with a negative circumstance, so it leads to withdrawal, disengagement from the world, and intense thinking to re-plan, shift perspectives and change his strategy.

Cognitive appraisal theories have formed the basis of a number of computational models of emotion [50, 133, 118]. These models serve three functions: First, they quantify and systematize the processes that are outlined in psychological theories. Rather than describe fear as a threat appraisal, they provide concrete definitions of when a stimulus is seen as threatening, quantify the intensity of the ensuing fear response, and precisely define how quickly the emotion will ebb. Second, they allow researchers to test the correctness of the theories they are based on. In controlled experiments, subjects' emotion expressions in realistic situations can be recorded and compared to what the model would predict. This comparison and contrast also allows for amendments to the theories, or the development of new ones. Third, high-fidelity computational models of emotion can be used as part of the design of a virtual agent, whose emotion expressions will appear "natural" and human-like to people interacting with it.

This thesis builds upon much of this prior work. In particular, it builds appraisal-based computational analogues of emotion functions. These computational analogues make use of the same appraisal variables that are widely employed by existing models. Rather than being aimed at describing human affective behavior in a high-fidelity

manner though, those computational analogues are deployed as part of a computer agent's design to improve its decision-making.

1.2 Background: Basic decision-making models

The techniques presented in this thesis address the problem of decision-making for a single agent, as well as in the context of a multi-agent system. Prior research in multi-agent systems has used a variety of techniques and algorithms for agent decision-making. These include decision theory, game theory, logic, reactive models, and Markov Decision Processes (MDPs), among others. This thesis draws on and contrasts its approaches to two of these, MDPs and game theory. This section sets the background for the rest of the thesis. For a detailed exposition on game theory, MDPs and other techniques the reader is referred to introductory textbooks such as Myerson's [88] and Luger & Stubblefield's [79].

1.2.1 Markov Decision Processes

Markov Decision Processes (MDPs) and their variants are being introduced below in some detail, as they pertain to some of the issues discussed in Chapters 2 and 3. An MDP serves to analyze long-running decision-making problems in an uncertain, stochastic world. At every point in time $t = 1, 2, \ldots$, the world is in state $s^t \in S$, where S is the set of all possible states. A single agent chooses an action a^t from a set of possible actions A in every timestep t. This results in a one-time reward that depends on the chosen action and the state of the world $u^t = u(a^t, s^t)$. It also results in the world possibly transitioning to state s^{t+1} according to a probability transition

function τ^t, such that $\tau^t(s^t, s^{t+1}, a^t)$ is the probability that s^{t+1} will be the next state of the world if action a^t is taken in state s^t at time t. *Time-invariant* MDPs are most commonly used, in which $\tau^t = \tau, \forall t$; in other words, the world's stochasticity is governed by the same rules across time. Another restriction typically placed on the transition function is *ergodicity*. Informally, an ergodic transition function has the property that, from every state $s \in S$ it is possible to directly or indirectly transition to any other state $s' \in S$ in a finite number of steps. An *policy* π for a time-invariant MDP prescribes an action $\pi(s)$ for the player to take in every state s. An optimal policy has the property that following it gives the agent the highest expected long-term reward over time periods $1, 2, \ldots$, in which future payoffs are exponentially discounted. Thus the optimal policy maximizes $E[\sum_{t=1}^{\infty} \gamma^{t-1} u(\pi(s^t), s^t)]$, where $\gamma \in (0, 1)$.

Variants of this simple framework are often used: In Multi-Agent MDPs (MA-MDPs) the world transitions based on the actions of more than one agent, and the optimal policy maximizes the sum of their utilities. In Partially Observable MDPs (POMDPs) the agent is uncertain about the state of the world s^t, and maintains a belief β as a probability distribution over what the current state and all previous states have been. This belief is updated by gathering observations $\omega^t(s^t)$ from the world in every time step, which are correlated with the true state of the world s^t. Finally, in Decentralized POMDPs (DEC-POMDPs) the agents receive different observations $\omega_i^t(s^t)$, and have to reason about others' observations. They may also choose to explicitly communicate and share information to better coordinate their efforts.

1.2.2 Game theory

Game theory is an economics-based approach. It treats decision-making as a maximization problem. The agent's goals and preferences are described by a utility function, such that higher-utility outcomes are preferred over lower-utility ones. The agent then acts by choosing the actions that maximize its utility function, by also considering how other agents will behave.

Game theory analyzes situations as *games*, which are mathematical constructs designed to describe, predict or prescribe the behavior of intelligent rational actors. In a game, actors (people or agents) are called *players*. They act by selecting a *strategy* that prescribes what they will do under every possible contingency that might arise in the game.

More specifically, in its simplest form a game consists of: (a) a set of players $N = \{1, 2, \ldots, n\}$; (b) a set of pure strategies S_i for each agent $i \in N$; and (c) utility functions $u_i : \times_i S_i \to \mathbb{R}$, one for each player, that map a choice of pure strategy for each player to a real number, representing the utility of player i if every player were to choose those strategies.

A simple example will illustrate this terminology: Consider the oft-mentioned Prisoner's Dilemma game, in which two business partners have been arrested for embezzling money. They are interrogated in separate rooms and are both given the same choice. If they provide evidence for their partner's guilt, they will get lenient treatment and walk out without any penalty, but their partner will be sentenced to 10 years in prison. Vice versa, if they keep silent and their partner betrays them, they will get 10 years in jail and their partner will walk free. If they both speak their

testimony has less value, so they will both get a 2-year conviction. If, however, they both keep silent, there will be evidence to only convict them for half a year. How can we expect them to behave in this situation?

Game theory offers a systematic way to answer this question. The two players in the game are $N = \{1, 2\}$. Each has a strategy set consisting of 'keeping silent' (K) and 'betraying' their partner (B), so $S_1 = S_2 = \{K, B\}$. Their choice $s_i \in S_i$ yields utility according to the penalty structure described above. For instance, if (s_1, s_2) is the vector denoting both player's choice of strategy, player 1's utility function has $u_1(K, K) = -0.5$, $u_1(K, D) = -10$, $u_1(D, K) = 0$ and $u_1(D, D) = -2$.

This is called a game's "Normal form" representation. For games with two players, the Normal form can be conveniently represented as a table, in which the two players assume the "row" and "column" position, respectively. The table has one row for each pure strategy of player 1, and one column for every pure strategy of player 2, and each cell of the table represents the utilities obtained by both players if they were to select the strategies corresponding to that row and column. In Table 1.1 below, the Prisoner's Dilemma game is represented in tabular form.

	Keep silent	*Betray*
Keep silent	(-0.5, -0.5)	(-10, 0)
Betray	(0, -10)	(-2, -2)

Table 1.1: The Prisoner's Dilemma game

Having represented the situation as a game, we can now solve it to predict what rational intelligent agents would do. The most popular solution concept in game theory is called a *Nash equilibrium* [90]. A Nash equilibrium is a "fixed point" in

the strategy space, in the sense that, if all players chose strategies that together form a Nash equilibrium, no player would gain by unilaterally deviating to another strategy. The Nash equilibrium therefore enjoys the property of stability. Moreover, it represents a choice by intelligent rational actors. In a Nash equilibrium, every player is optimizing his or her behavior given what all other players are choosing to do.

Going back to the Prisoner's Dilemma example, consider player 1. If player 2 chooses K (keep silent), it is in player 1's benefit to betray him (play B) and gain a utility of zero, as opposed to -0.5. If player 2 chooses to betray him (B), again it is in player 1's benefit to also betray him, since that given him a utility of -2 as opposed to -10. The situation for player 2 is exactly symmetrical. Hence, both players will rationally choose to betray each other, making the point (B, B) in the strategy space a Nash equilibrium of the game. To see why, consider the stability property: for both players, given the fact that their partner is going to betray them, their maximum utility is obtained by also betraying him.

Game theory allows significantly more complex situations to be modeled in the same way. Games can include more than two agents, and these agents need not act simultaneously or take just one action each. For instance, in the popular game of tic-tac-toe the players alternate until one of them wins or the game ends in a tie. The players may also randomize when they choose strategies. A *mixed* strategy is a probability distribution over a player's pure strategies. For example, in the Prisoner's Dilemma, player 1 may play the mixed strategy that will keep silent with 70% probability and betray the other player with 30% probability. Every game is

guaranteed to have a Nash equilibrium, but in some games this equilibrium might be in the space of mixed strategies. Games may also have more than one equilibrium. Multiplicity of equilibria is especially common in *repeated* games, in which a one-shot game, such as the Prisoner's Dilemma, is repeated between the same players for a number or rounds (possibly infinite). In those cases, the strategy of a player for the overall game dictates a choice of strategy for each repetition of the one-shot game. In the case of multiple equilibria, certain equilibrium selection criteria exist to select which equilibrium rational players may be expected to implement in practice, such as Pareto efficiency.

To identify Nash equilibria of games, many algorithms have been developed. For certain classes of games, such as 2-player, zero-sum games (in which the payoffs of both players sum to zero for every choice of strategy), this can be done efficiently in polynomial time [22]. In the general case, the equilibrium computation is believed to require exponential time [28]. Nisan et al.'s textbook on algorithmic game theory [97] provides an excellent discussion on the computational challenges of Nash equilibrium.

Other representations have been used to either make the size of the representation more manageable, to provide better insight to a human about what is happening in the game, or to more efficiently capture interesting situations. The *extensive form* of a game is an inverted tree structure. Nodes represent decision points in which a player must act, and edges represent actions taken. The extensive form allows the time aspect of games to be more explicitly represented, as the tree describes which player goes first, who follows, and what actions are available to them at each point. It also highlights uncertainty, which is known in game theory as *imperfect information*.

For instance, in the Prisoner's Dilemma, player 2, when choosing whether to keep silent or betray his partner, does not know player 1's choice. In the tree, this is explicitly denoted by grouping the two cases (player 1 kept silent, player 1 betrayed) into a single "information set," as shown in Figure 1.3.

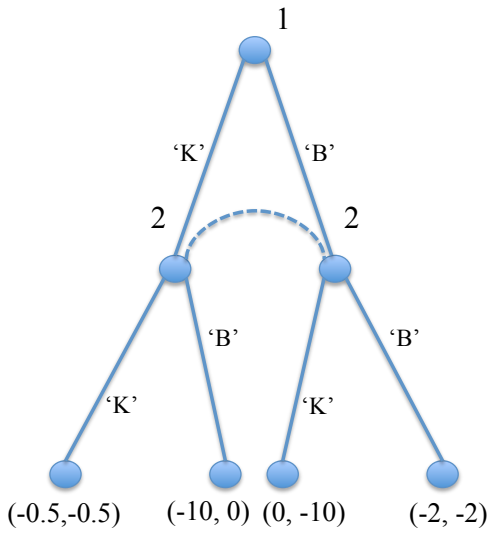

Figure 1.3: The Prisoner's Dilemma in extensive form

Other representations help reduce the size of a game, by exploiting its structure. Graphical games [65] are useful in games in which the utility of some players only depends on the actions of a subset of other players. Action-graph games [61] simplify cases in which the utility of each player depends on the number of other players that choose a particular action, but not on who they are. Multi-Agent Influence Diagrams, which are extensively described in Chapter 5, allow more efficient reasoning about the stochasticity of the world and the players' uncertainty.

1.3 Structure of the thesis

Chapter 2 presents GRUE, the emotion-inspired computational operators that are used to re-prioritize a single agent's goal, along with empirical findings demonstrating the usefulness of the approach. The affective signaling mechanism for inferring unobservable characteristics of other agents is discussed in Chapter 3. In Chapter 4 a human-computer interaction study is presented, in which the influence of emotion expressions on people's perceptions of the agents' trustworthiness is investigated. Chapters 5 proceeds to discuss the game-theoretic contributions of this thesis that build upon the theory of reasoning patterns, while Chapter 6 focuses on deploying of the reasoning patterns toward helping human decision-makers. The thesis concludes in Chapter 7 with a summary and extensions for future work. Bibliographic references can be found in the end of the dissertation.

Chapter 2

Re-prioritizing Goals Using Affect-Like Computational Operators

The environments in which computer agents act are becoming increasingly complex. This complexity manifests as the expanding size of the domain, the increased stochasticity of the world, and the high levels of uncertainty the agents have to work under. This chapter discusses the challenges complex environments pose for agents' decision-making, and some of the tools and techniques developed to address them. It then presents the use of affect-inspired computational operators to address these challenges.

The structure of the chapter is as follows: Section 2.1 discusses the ways in which *scale*, *hardness* and *uncertainty* increase complexity, making decision-making more difficult for agents. Section 2.2 presents a new method, GRUE (Goal Re-prioritization

Using Emotion), which was inspired by certain beneficial cognitive features of emotions (see Section 1.1). GRUE uses computational operators analogous to human emotions to re-prioritize an agent's goals. These operators are defined in a domain-independent manner and are triggered automatically, without the agent having to explicitly reason about which ones it should be using at any point in time. GRUE offers a way for an agent to make decisions quickly, respond efficiently to external events, and to a large extent circumvent the issues associated with high complexity. Section 2.3 presents other related work in AI around the issue of complexity, while Section 2.4 includes a short discussion of GRUE.

2.1 Complexity in Decision-Making

Computers systems are used in a large number of settings in which decision-making is required of them. Two examples will be used throughout this section to illustrate various aspects of computer decision-making. These examples are drawn from opposite extremes of the complexity spectrum. The first, representative of relatively simple domains, is a recommender system that allows users to rate movies they have seen, and then uses these ratings to recommend other movies that they might also enjoy. The second, representative of more complex environments, is a self-driving car that must navigate itself through the streets of a city quickly, safely and lawfully.

In general, computers making decisions are expected to satisfy a number of requirements. First, the decision must be *of high quality*. The quality of a decision reflects the degree to which it brings about states of the world that are desirable to

the computer agent. Second, a decision must be made in an *efficient* manner. Efficiency relates to the resources (time, space, communication) required to compute it. Third, the decision-making performance of the agent must be *robust*. It should be expected to perform reasonably well in all situations, and avoid significant mistakes even in rare and uncommon circumstances, or at least degrade gracefully in the face of unexpected inputs.

2.1.1 Scale, Hardness and Uncertainty

Real-world environments present three challenges to traditional computer decision-making techniques in Artificial Intelligence. First, there are theoretical and practical issues stemming from the sheer *scale* of some domains. Second, most of the widely used solution concepts, such as Nash equilibrium, have been shown to be *computationally hard*, requiring exponential time. And third, aspects of the world might be unobservable, or the environment might be stochastically changing in ways that are not fully known to agents, increasing their *uncertainty*. The term "complex environment" shall be used to refer to any domain that is characterized by these three features.

Scale

Agents making decisions in an environment have to acknowledge that the world around them is constantly changing. To represent this change, the world is assumed to be in some *state* at any given point in time, and transition between those states as time flows and agents are taking actions. In a small system the number of these

states is small and manageable. Moreover, the agent usually has a small set of actions to choose from. Also, if the agent is uncertain about some quantities in the world (e.g., the value of a hidden variable, the state of an unobservable part of the world or the private information of another agent), these things are usually few in number and a probability distribution capturing the agent's uncertainty can be compactly represented, maintained and updated over time. Furthermore, even in larger domains, certain assumptions about the dynamics of the world are typically made that simplify representing the environment. For instance, it is usually assumed that the world operates under 1-step Markovian dynamics. This in most cases relieves the agent of having to track an extended history of changing world states—it only needs to reason about the world's present state.

In many real-world domains, however, these convenient assumptions no longer hold. The world itself becomes larger, and the number of possible states may rank in the order of millions or billions. This fact alone complicates agent design. For instance, to capture a single probability distribution over a billion states one needs a floating-point vector of size one billion. Maintaining such a vector in memory and updating it is an expensive operation in terms of both time- and space-complexity. Furthermore, in many environments the agent might have a large action set to choose from. Finally, if Markovian assumptions no longer hold in the world, an extensive history of observations must be maintained to effectively track the state of the world, predict future developments and respond with appropriate actions.

The effect of scale differs in the two sample domains. In recommender systems each movie can be described by just a few features: year of production, genre, director,

acting cast, etc. On the other hand, a self-driving car might operate in a much richer world, in which it can potentially register anything from the shape and incline of the road, water puddles, pedestrians, traffic lights, road signs, other cars, to the slightest reading of any of its sensors. The sheer size of the world makes fully representing it intractable, and severely limits the type of algorithms and solution concepts that may be used to make decisions, as explained below.

Hardness

There are both theoretical and practical challenges linked to the *computational hardness* of most popular solution concepts, such as Nash equilibrium. On the theory side, Nash equilibria have been placed in a class of problems (PPAD#) that is widely believed to require exponential time to solve [28].

In practice, several issues in equilibrium computation arise from the implicit assumptions that equilibria carry. In particular, game-theoretic analyses typically make demanding epistemic assumptions, such that every agent knows everyone's utility function, action set, and the structure of the game. Even if uncertainty exists about some of these quantities, this uncertainty is usually modeled as a common, publicly-known probability distribution, and the definition of Nash equilibrium is extended to reflect the agents' uncertainty (Bayes-Nash equilibrium). However, in many realistic settings it would be unreasonable to assume that all agents possess identical prior beliefs (see Morris's discussion on the epistemic validity of the common prior [86]). As soon as the common prior is abandoned, however, the very idea of Bayes-Nash equilibrium is called under question. Although Bayes-Nash equilibria can be defined

for games without a common prior, its absence makes these equilibria "private," in the sense that every agent computes different equilibria based on its own prior [2]. Thus, the equilibria computed by one agent are not very informative about how it expects other agents to actually behave during the game, even if the fundamental assumption that they will follow some Bayes-Nash equilibrium is maintained. These non-common priors also result in a computational problem. Computing an appropriate course of action without a common prior requires taking into consideration *belief hierarchies* like "agent 1 believes that agent 2 believes that agent 3 believes that agent 1 believes..." that rapidly make equilibrium computation a convoluted problem, for which no particularly good approach exists.

Our two sample domains help illustrate this point. The recommender system faces no significant problems with strategic reasoning, because people rating movies have no reason to lie about which movies they liked. In the self-driving car example, on the other hand, coordination—and thus equilibrium concepts—is crucial, as cars and pedestrians should ideally wish to best-respond to each other's actions to avoid collisions yet move effectively through space. Given the scale of the domain and the possibly inconsistent beliefs the different parties might have, the hardness of computing these solution concepts makes standard game-theoretic approaches intractable.

Uncertainty

Uncertainty stems from a variety of sources: One is a lack of complete knowledge about the world, as in the cases of incomplete information and non-common priors described above. Another is the fact that, although the world might be observable, no

good and accurate model may be known regarding its stochastic dynamics, making predictions about how it might change in the future hard. Probabilistic reasoning techniques typically assign a probability distribution for every uncertain quantity. There also exist methods for tracking the change of partially observable or unobservable quantities across time. These methods work by assuming some prior knowledge about the transition model of these quantities, and use subsequent observations to infer about how that variable might have changed over time.

In rich environments, however, it is often impractical to represent uncertainty as a probability distribution. One reason is that the possible states of the world are too numerous to account for individually. Another reason is that we might have no idea whatsoever about the likelihood of a variable's possible values, and we might not feel confident assigning it a uniform prior. A third reason has to do with the computational intractability of tracking variables across time. In simple settings with just a few variables and simple world dynamics this is fairly straightforward, but in slightly more complex domains even approximations are sometimes provably impossible [101].

Consider the two sample domains: The unobservable hidden quantity in a recommender system is primarily the genre preferences of the different viewers. This quantity is not changing very rapidly—if at all—and can be inferred given adequate user input. On the contrary, the algorithm in the self-driving car must continually monitor the changing the state of the world as it moves along the road, as well as any (more or less) unexpected events that might have to be taken into consideration. The algorithm must maintain likelihoods for any of a huge number of possible events, such

as pedestrians crossing the street, bumps on the road, sudden moves by other cars, etc. Furthermore, many of these unknown quantities are not discrete but continuous, making maintaining belief distributions and performing inferences harder.

2.1.2 Deliberative and Reactive Methods

Researchers in AI have addressed the problems arising out of complexity in various ways, a brief overview of which is presented in Section 2.3. Some of these approaches try to take advantage of the structure of the world, by leveraging independence assumptions (e.g., Bayesian networks), using more efficient data structures (e.g., hierarchical models), or processing information in better ways. Most of these methods aim at speeding up decision-making, but wish to preserve the optimality of the solution they return. On the other hand, some other approaches are willing to sacrifice some optimality guarantees in order to reduce computation load and execution time (e.g., heuristics and abstractions). As the scale and the uncertainty in a domain increase, however, optimality is increasingly difficult to achieve, and even approximations have been in some cases formally shown to be intractable to derive [101].

In light of this, one should consider both *deliberative* and *reactive* techniques to deal with complexity. All decision-making methods can be placed in a continuum between those two extremes. For example, algorithms that identify Nash equilibria are fully deliberative methods, as they consider everything that is known about the world and other agents, and reason about all possible action plans of the agent, all contingencies that might occur in the world, and all the ways other agents might behave. On the other hand, other approaches have a strong reactive flavor. A myopic

chess-playing algorithm is reactive, as it essentially reacts to the actions of its opponent, by using a local scoring function to evaluate the current-step "goodness" of all its own candidate actions, but without reasoning about their long-term consequences.

The distinction between reactive and deliberative approaches is compelling and useful because it reveals which part of the continuum might contain the algorithm that will work well in a particular domain as a function of that domain's complexity. At one extreme, consisting of simple, small-scale, largely deterministic environments, one may use a fully deliberative approach. At the other extreme, if little is known about the domain, and the world behaves very stochastically, a heuristic that uses local criteria to select an action might perform better. And between those two extremes there is a progressive change of focus. As scale and stochasticity increase, not only do fully deliberative algorithms scale poorly, but they face two more issues. First, in trying to learn how the world works, algorithms in complex domains might lack the volumes of data required to learn a good model of the world. Second, if the world is very stochastic, by the time an algorithm has deliberated to compute an optimal (or near-optimal) solution, this might no longer be optimal (or even satisfactory) given the changes in the world that have ensued in the meantime. Hence, in those settings it might be of critical importance to be able to *adapt* reactively and successfully to what is happening in the environment, rather than compute the optimal solution a priori.

2.2 GRUE: An Emotion-Inspired Method for Decision-Making

The challenge of complexity in decision-making is being approached in this thesis by considering specific facets of the way people's emotions contribute positively to their decision-making. Intuitively, one can think of the method as a "smart" reactive algorithm, or as a general-purpose, domain-independent class of heuristics. The agent in GRUE does not attempt to track the complete state of the world or compute an expected utility function over time. It thus avoids the problems outlined for fully deliberative algorithms. Instead, at every time step, the method makes a decision by maximizing a valuation function myopically, ignoring the long-term consequences of the action to be taken. The "smart" part comes from emotion-inspired computational operators, defined below, which serve to update the weights in that valuation function by looking at the end-result (payoff obtained) of past decisions made. Changing the weights alters that valuation function to reflect changing circumstances in the environment. The method is thus called 'GRUE,' short for 'Goal Re-prioritization Using Emotion.' Schematically GRUE is depicted in Figure 2.1.

Evolutionary psychologists have argued that people's emotions have evolved to take advantage of statistical and structural regularities in the world, offering broad good responses to largely similar situations. For instance, most situations in which an important goal of a person is threatened serve to elicit 'fear' in people. The presence of fear in turn leads to responses that are generally successful against danger, such as withdrawing, fleeing, seeking shelter, or asking for help. As such, the emotion of fear

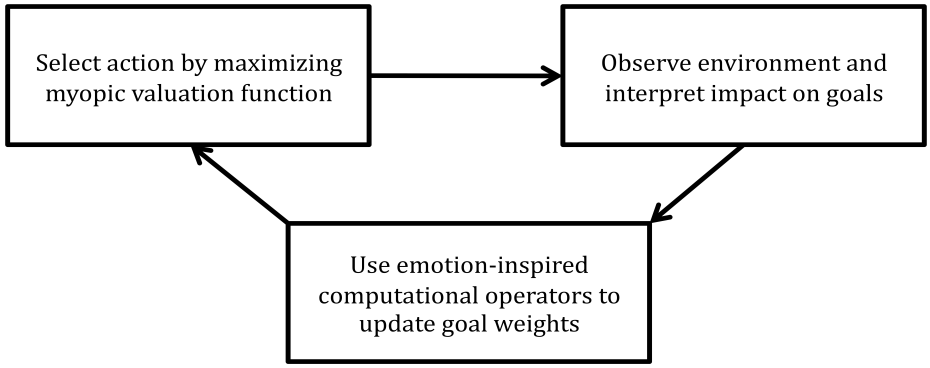

Figure 2.1: The GRUE method for decision-making

is then a heuristic that behaviorally groups a large number of different circumstances using a simple criterion (is it dangerous or not?) and offers a set of adaptive responses that have a high likelihood of addressing the problem. In doing so, emotions implicitly recognize important independencies and regularities in the domain. For instance, fear and its associated behavioral repertoire recognize that danger is a cue towards ignoring a large part of the state and action space, and instead focusing one's computational resources to percepts, facts, beliefs and actions that will help the person flee or evade that danger.

In summary, GRUE assumes that agents have goals they are trying to accomplish. Goals represent what the agent is designed to perform in the environment. For instance, a robot exploring the surface of Mars has to locate interesting-looking rocks, collect samples, avoid falling, and return to base, and these are its goals. It must be noted here that the term "goal" has been used in the literature to refer to many different concepts. In psychological theories, goals reflect the biological and social needs of people, such as obtaining food, mating, or protecting oneself. In AI, goals

have been used in logic-based frameworks to refer to attributes or states of the world that the agent is trying to bring about. In this chapter the term "goals" is used in a strictly technical sense to refer to the sources of an agent's utility. The goals of the Mars-exploring robot, which are to locate and collect rocks, stay standing, and return safely to base, are the events that yield positive rewards for it.

The existence of goals allows one to meaningfully represent emotions. Psychological theories of cognitive appraisal assume that emotions are generated by interpreting the impact of observations on someone's goals. For example, fear at the sight of a dangerous animal is generated by interpreting the animal's presence as a threat to the one's survival. Moreover, goals offer a principled way to represent the agent's preferences. In economic-based approaches a utility function is typically used to do this and the agent maximizes its expected utility over time. In GRUE, the agent is not maximizing expected utility, as that might be too difficult to compute. Instead, it myopically maximizes a weighted valuation function in every time step. Because the maximization is myopic, the agent can make decisions very quickly. GRUE's core contribution, however, is that the weights in that valuation function allow the agent to adapt to changing circumstances. In particular, computational operators that resemble aspects of human emotions are constantly and automatically monitoring what happens in the environment. When an operator like that "fires," it alters the weights in the agent's valuation function. As a result, the agent is not maximizing the same quantity in every time step, but essentially "switches focus" among its goals. Specifically, when a goal's weight increases, the output of the myopic maximization is likely an action that promotes and caters to that goal. Hence the agent can be thought

as reactively adapting its decision-making focus to developments in the world. The emotion-like operators carry important insights from the way people have evolved to switch focus into the design of computer agents.

Goals and valuation

An agent is designed to perform certain functions in a particular environment. These functions of an agent can be thought of as its *goals* (G). In GRUE, each goal $g_i \in G$ is associated with three numbers:

First, each goal has a value $v_i \in \mathbb{R}^+$, denoting its relative importance. This value is considered domain knowledge and is not changing across time. For example, the value of collecting a sample in the case of a Mars robot might be less than the value of maintaining good system status and communication.

Second, every goal has a priority $r_i^t \in \mathbb{N}^+$ in every time step $t \geq 0$. Goals of higher priority (larger r_i^t) are considered more significant in the current time step, whereas goals of lesser priority are considered less significant. While the value v_i of a goal does not change over time, its priority might differ between time steps.

Third, each goal has a degree of achievement $d_i^t \in [0, 1]$. This is a measure of the agent's subjective expectation that the goal will be achieved at some point in the future. A goal with $d_i^t = 0$ is forever lost (and hence $d_i^t = 0$ implies $d_i^\phi = 0, \forall \phi \geq t$. A goal with $d_i^t = 1$, similarly, is considered accomplished. Goals with intermediate values in their degree of achievement are more or less closer to being achieved. In a sense, the quantity d_i^t is a simple measure of how well the agent is doing with respect to goal g_i.

These three figures (value, priority and degree of achievement) combine to give a valuation function

$$\hat{u}^t = \sum_{g_i \in G} d_i^t \cdot q(v_i, r_i^t)$$

As can be seen in the above formula, the valuation at time t is proportional to the degree of achievement of all the agent's goals. In other words, if the agent is taking actions to maximize its valuation function, it is essentially acting to increase the degree of achievement of its goals. However, the degree of achievement of each goal is weighted by that goal's value and current-step priority, in a quantity denoted by the $q(\cdot, \cdot)$ function. In the experiments that were run it was stipulated that $1 \leq r_i^t \leq r_{max}$ and $q(v_i, r_i^t) = v_i \cdot (r_i^t / r_{max})$ was used as the q-function, although arguably other ways to weigh value and priority against each other are possible (for instance one could have defined q to be equal to $(1 + v_i)^{r_i^t}$).

Since goals are weighted by their priorities, by sufficiently increasing the priority of a goal the agent will be geared towards taking actions to accomplish it rather than lesser-priority goals. Changing the vector $\mathbf{r}^t = (r_1^t, \ldots, r_{|G|}^t)$ provides a way to change an agent's valuation function across time. This change induces adaptive behavior, by essentially switching the agent's focus between its various goals. Notationally, $\hat{u}(\mathbf{r}^t, \mathbf{d}^t)$ will be used to denote function $\hat{u}$ parametrized by priority and degree of achievement vectors $\mathbf{r}^t$ and $\mathbf{d}^t$, respectively.

The hat operator ($\hat{}$) serves as a reminder that this is not the agent's *actual* utility. This other function $(u(a^t, s^t))$ denotes the reward the agent will obtain by taking action a^t in state s^t of the world. It therefore captures the agent's true preferences and is not changing from time step to time step. However, computing the long-

term expected utility $\sum_{s^t,s^{t+1},\ldots} u(a^t, s^t)$ might be too complex to even represent, as it needs to incorporate every action taken and every event to occur across the agent's lifetime, interactions between events, costs and probabilistic dependencies, among other factors, all of which influence state transitions. Moreover, it requires that the agent explicitly reasons about the state of the world, which might not fully observable, or whose transitions might be governed by dynamics unknown to it. As a result, such a function might therefore be too complex or impossible to even write down. In contrast, $\hat{u}^t$ comes merely from the functional specifications (desired behavior) of the agent at every point in time. Agents in subsequenly-described simulations will make decisions by maximizing $\hat{u}^t$, but their performance in the end of the simulation will be evaluated with respect to their actual utility function u.

Optimizing the valuation function and state-keeping

To maximize the valuation function $\hat{u}(\mathbf{r}^t, \mathbf{d}^t)$ at time t the agent must possess some domain knowledge. First, it needs to be aware of the action set A available to it. Second, for every action set $a \in A$, it must have some notion of how that action will affect the degree of achievement of its various goals. For instance, the Mars-exploring robot in our example might have actions such as 'pickup item,' 'walk forward,' 'turn right,' etc. It should also know that if there is a chasm ahead, 'walk forward' will probably result in the degree of achievement of goal 'stay healthy' to drop significantly.

We thus offer our agent another function, $f(\mathbf{d}^t, a)$. This function receives as arguments the degree of achievement vector $\mathbf{d}^t = (d^t_1, \ldots, d^t_{|G|})$ and a candidate action

$a \in A$, and returns the expected degree of achievement vector *after* the action has been taken. To accomplish this, the function f must incorporate domain knowledge. This domain knowledge is assumed to be rudimentary, and not necessarily accurate or fully quantified. In other words, as d_i^t is only an estimate of how well the agent is doing with respect to goal g_i, thus the quantity $f(\mathbf{d}^t, a)$ is only an estimate of the impact of action a. All that matters is that f vaguely captures the direction (positive or negative) and relative magnitude of a's impact on the agent's goals.

The function f, as defined above, is *stateless*. This is however not very useful in practice. For example, 'walk forward' at the edge of a chasm is negative to the agent's well-being, but otherwise it might be positive. To discriminate between such situations the agent might maintain a state $\sigma^t \in \Sigma$ for the current time step t, and then condition the impact of an action on that state ($f(\mathbf{d}^t, a|\sigma^t)$). Again, σ^t does not necessarily reflect or correspond to the full state of the world at time t, which is denoted by s^t, as that might be unobservable and very difficult to accurately track. The state-space Σ is simply part of the agent's design and only serves to discriminate between situations in which an action might elicit very different effects. The agent also makes no attempt to track or predict the future transition of σ^t (or of s^t).

Given this function f, we have a way of selecting an action in every time step: the agent simply chooses action

$$a_t^* \in \text{argmax}_{a \in A} \hat{u}(\mathbf{r}^t, f(\mathbf{d}^t, a|\sigma^t))$$

In other words, the agent selects the action with the highest expected increase in the degree of achievement of its goals, each weighted by its value and current priority. In maximizing its valuation function, the agent is treating goal priorities as constants,

i.e., it does not reason about how these might change in the future because of its actions.[1]

The agent is, however, also able to *observe* things in the environment. Things like the presence of a threat or a reward ought to change its beliefs regarding the degrees of achievement of its goals. To incorporate this functionality, we add the observation function $b(\mathbf{d}^t, e)$, where $e \in E$ is an observation obtained and E is the set of all possible observations. The function returns—much like f—an updated degree of achievement vector. For instance, if a system error is suddenly detected, the function b will return a degree vector in which the d_i of goal "maintain system health" is decreased. At every time step t an observation e^t is received and

$$\mathbf{d}^{t+1} \leftarrow b(\mathbf{d}^t, e^t)$$

Also, the observation e^t also serves to update the agent's state

$$\sigma^{t+1} \leftarrow \psi(\sigma^t, e^t)$$

The function ψ is a simple state-update function. In the simulations, it merely assumes that σ^{t+1} is the same as σ^t insofar as the observation e^t does not contradict it. If any elements of e^t are incompatible with σ^t, those elements alone in the next-step state σ^{t+1} are amended to reflect this. For instance, if σ^t denotes the presence of a chasm ahead, and e^t reports no chasm, the state σ^{t+1} will also not report a chasm.

Any domain-specific knowledge the agent possesses is contained only within the values of goals v_i and within the functions f, b and ψ. Furthermore, these are directly

[1]This is an important way in which GRUE differs from established techniques, such as MDPs. In an MDP, for example, actions are selected by maximizing the Q-value of a state-action pair. This Q-value, however, has already been computed to capture information about the long-running effects of this choice under the policy followed by the agent. In GRUE, future effects of an action are only captured by goal priorities, which are computed and adjusted reactively.

derived from the functional specifications of the agent design and are unavoidable in every agent architecture. For instance, a typical game-theoretic decision-making algorithm (say, a POMDP) would still need to somehow define a utility function for the agent, and reason about the predicted effects of actions and the interpretation of observations. The fact that domain knowledge is limited to the v_i's and the functions f, b and ψ allows the emotion-like operators, presented below, to be defined in a domain-independent manner.

Emotion-inspired operators

The function of the emotion-inspired computational operators is to update the priority vector $\mathbf{r}^t$. These computational operators are inspired by the beneficial functions of human emotions, but they are not meant to fully replicate these emotions as they manifest in people. The functional characteristics of the operators are therefore "similar" to those of people's emotions, but they do not serve as definitions of emotions in the psychological sense. Operators are of two types: *Goal-specific* operators are elicited in connection to, and update the priority of a particular goal. For example, fear is elicited when a particular goal is threatened and impacts the relative priority of that goal. *Goal-independent* operators are elicited without a particular goal in mind and may change the entire priority vector. For instance, the operator of boredom is elicited when nothing out of the ordinary happens for a while, but is not associated with any particular goal.

Each emotion operator $m \in M$ is associated with two functions. On the one hand, there is the *activation* function λ_m, which contains the elicitation condition for the

emotion and returns TRUE if that condition is met. On the other hand, there is the *consequence* function κ_m, which describes how the priority vector is to be changed when the emotion is present, i.e., when λ_m becomes true. Below the functional form for some emotions is presented:

- Hope and fear are reactions to changes in the expectation of the future accomplishment of a goal, positive or negative (hence, hope and fear are goal-specific emotions). Both hope and fear in people are emotional reactions to a perceived *change in expectation*. Hope is evoked, essentially, by realizing that something desirable is more likely to occur than originally believed. Fear, similarly, is evoked by becoming aware that something has just come under threat. Both these realizations lead to a shift in focus and motivation. Fear leads individuals towards protective and defensive actions like seeking shelter or hiding. Hope, on the other hand, tends to establish the success of currently-followed behaviors with respect to the goals being hoped for. Hence, fear causes the person's focus to shift to the goal being feared for, in order to protect it, while hope has the opposite effect, since the goal that is faring better than expected needs no urgent intervention. Since these two emotions have similar characteristics, similar functional definitions are being provided for both of them. We define λ_m for fear to be TRUE iff $d_i^{t-1} - d_i^t \geq \theta_1$, where θ_1 is an externally set threshold. For hope, this condition becomes TRUE iff $d_i^t - d_i^{t-1} \geq \theta_1$. In essence, fear and hope are elicited when there is a significant change in the expectation that a particular goal g_i will be accomplished. For instance, if an system error shows up, function b will cause a decrease in the d_i for goal "avoid enemy" and this

will elicit fear (if larger than θ_1). We also define κ_m to increase (or decrease) the r_i^t of the threatened (or hoped-for) goal by a constant c. This change, if sufficient, will direct the agent's efforts into protective actions, such as running away, or allow it to tend to the needs of other goals.

- Boredom, a goal-independent emotion, is elicited when the $\hat{u}$ experienced by a number of rounds has not changed significantly. In particular, λ_m is TRUE iff the standard deviation of payoffs $\{\hat{u}^{t-\psi}, \ldots, \hat{u}^{t-1}\}$ does not exceed a certain threshold θ_2 (in the simulation ψ, the length of history considered for the elicitation of boredom, has been set to 10). When activated, the emotion of boredom perturbs the priority vector $\mathbf{r}^t$ at random. The intuition behind this definition is that boredom detects situations in which the agent is "stuck at a local maximum," i.e., situations in which its behavior has historically lead to good payoffs, but which might prevent it from exploring even better alternatives in the behavioral space.

- Anger is an emotion that gets elicited when blameworthiness can be attributed to another party for some negative-impact event according to the OCC model of emotions [100]. As such, its elicitation λ_m must consider whether an observed change in the environment has been caused by some other agents, and whether they had an alternative course of action or they specifically intended to cause it, as opposed to it being a side-effect of their plans (to establish blameworthiness). Anger results in the raising of priorities among goals geared toward harming the supposed perpetrators or negating the effect of their actions. As such, it accomplishes the basic functions attributed to the emotion of anger in

people, which are (a) to identify others who have acted maliciously or broken accepted social norms, and (b) to offer a punishment mechanism that deters such behaviors and promotes cooperation.

- Sadness is an emotion that is elicited from repeated or significant failure. In humans it elicits withdrawal from activity and rigorous thinking, aimed at re-planning one's course of action. (In some cases this emotion might be employed maladaptively, causing chronic withdrawal, i.e., depression.) Consistent with this function of sadness, λ_m was set to be TRUE when a series of valuations $\{\hat{u}^{t-\psi}, \ldots, \hat{u}^{t-1}\}$ all lie below a certain fraction $\delta \in (0, 1)$ of the historically average or expected valuation, where $\hat{u}^t = \sum_{g_i \in G} d_i^t \cdot q(v_i, r_i^t)$ was the valuation function value at time t. The result of sadness is to suppress the priority of higher-priority goals, and increase that of low-priority goals, essentially "switching focus" to a potentially more promising set of actions.

The above emotion-like operators are defined in a general-purpose and not domain-specific manner. In the definition of each operator there is no specific features of the particular domain are mentioned. Instead, the operators' functional specifications make use of only cross-domain concepts such as $\mathbf{d}^t$, $\mathbf{r}^t$ and $\hat{u}^t$. These quantities serve as abstractions, as they may have a different interpretation in each domain, yet they allow the definition of the emotion-like operators to remain the same across domains. As shown in the evaluation, also, the performance of GRUE is very robust with respect to the threshold values chosen for the various operators.

To keep things simple, in the evaluation of GRUE only the emotion-like operators of hope/fear and boredom were implemented. Furthermore, the same threshold θ_1

for hope and fear was used in each case.

2.2.1 Evaluation of GRUE

Two domains were selected to evaluate GRUE that are characterized by high uncertainty and complex stochastic change: restless bandits and a foraging domain. The restless bandits are an extension of the problem of *multi-armed bandits*, which has been used extensively in AI as a prototypical example for the exploration/exploitation tradeoff. In the foraging domain, the agent exists in a "blocks world" and moves about to locate food. Friends and enemies also exist in the environment that will potentially benefit or hurt the agent, and there is no easy way of telling whether another entity is a friend or an enemy. Both domains have many desirable features that are common among complex environments, such as large scale, uncertainty, and provable hardness of computing traditional solution concepts.

In both domains, also, the same mechanism and the same emotion-like operators were used without change or adaptation, and only the functions f, b and ψ were specifically designed for each domain. This safeguards that the mechanism is not fine-tuned to the specifics of a particular environment. It also strengthens the argument that the particular emotion-like operators used might have potentially broader applicability to a variety of other domains.

Restless bandits

Restless bandits are an extension of stochastic multi-armed bandits (MABs). In a typical setting, the bandit consists of a number (k) of arms, each of whom delivers a

reward when selected. In the simple MABs case, the reward r_i of each arm i is drawn from a probability distribution that depends on the arm's state (s_i). In each time step, the agent may choose only one arm and obtain the reward from it; after this, the arm that was selected transitions stochastically to another state according to a transition matrix $\mathbf{T}_i$. The restless bandits case extends the above framework by allowing all arms to undergo a transition in each round, even those not selected. In particular, each arm i transitions to a different state in each round according to matrix $\mathbf{T}_i$ (if selected) or matrix $\tilde{\mathbf{T}}_i$ (if not selected). The goal in both cases is to maximize average reward. But whereas MABs admit an optimal solution, termed the "Gittins index" [48], restless bandits have been shown to be a hard problem. According to Papadimitriou et al. [101] even with deterministic transitions the problem is intractable in the general case. Recently, work has been done to be able to compute solutions for subclasses of the problem (e.g., Slivkins & Upfal [120]), most of which follow the "Whittle index." (For a good review and an approximation algorithm see Guha et al. [54].) Such solutions, however, suffer from assuming that too much is known: payoff distributions and transitions matrices, as well as the initial state of each arm, are usually considered known, and the problem is cast as being able to reap high rewards despite the state of all arms changing over time. However, this does not directly apply in situations where neither the stochasticity in payoffs nor in transitions is known, as in the setting here examined.

In the simulations reported below there were $k = 5$ arms, each of which could be in one of three states: "good," "medium" and "bad." In each state s of arm i, payoffs were given by a Gaussian with mean μ_i^s and standard deviation σ_i^s. Naturally, good

states had higher means than medium ones, which had higher means than bad ones.[2] An agent was allowed to make choices for a number R of steps (the value of which varied among 30, 100, 500 and 1500). In ever case the average payoff of the agent (as well as its variance) were recorded. For every agent tested, the experiment was repeated 100 times.

The emotion-based agent was given five goals, one for each arm, of the form $g_i =$ "obtain high reward from arm i." All goals had the same value $v_i = v$. The agent did *not* track the states of the various arms across time. It merely assumed that the state of every arm remained unchanged since the last time it was selected.[3] When the agent selected an arm, it compared the payoff received with historical payoffs from the same arm, and made a crude estimate whether its state was good, medium or bad. Given its assessment of the arms' states, action "select arm i" was expected (in function f) to increase the degree of achievement of goal g_i if arm i was believed to be in the good state, decrease it if i was believed to be in the bad state, and leave it unchanged otherwise. This is basic domain knowledge, merely stating that selecting good-state arms is better than selecting bad-state ones. After a reward was obtained, the degree of achievement of the corresponding goal would be adjusted accordingly (function b), increasing upon high rewards and decreasing upon low rewards ($d_i^{t+1} \leftarrow$ payoff received at t / max payoff). The agent employed the three aforementioned emotion

[2]This particular instance of the restless bandit problem belongs to the special cases that *are* tractable, in the sense that a Whittle index could be computed for them. Again, however, this would only be possible if the transition probability distribution were known; if not known, as in the experiments conducted, the optimal policy is, of course, impossible to compute.

[3]Since the state-space S in the restless bandit simulation was small (five arms in three possible states each, for a total of 15 states), the internal state-space representation for the agent, Σ, was taken to be the same as the actual state-space of the game, S.

operators of hope, fear and boredom to update the priorities of its goals, which were all initialized to 5 and were allowed to range between 1 and $r_{max} = 10$. Different values for the emotion thresholds were tried ($\theta_1 \in \{0, 0.1, 0.3\}$ and $\theta_2 \in \{0, 1\}$). Hence a total of six variants of the emotion-based agent were tested in order to examine the robustness of GRUE with respect to the choice of thresholds. The approach's performance was taken to be the average among the six variants examined in order to prevent any observed performance benefits being obtained by a fine-tuning of the threshold values.

The emotion-based agent was compared to six agents: (i) a random agent, which selected arms with uniform probability in every time step; (ii) a reactive agent, which selected an arm until a significant decrease (greater than 30%) in payoffs was observed, then switched to a different arm at random; (iii) a learning agent, which made assessments (based on past data) about the mean and variance of each state, but did not track the change in states across time; (iv) an "all-seeing" agent, who would (somehow) know the payoff means of all the states, as well as the current state of each arm; (v) a "half-seeing" agent, who would know the payoffs means of all the states but would not know their exact current state; and (vi) a collection of indexing agents. These indexing agents are guaranteed according to Guha et al. [54] to contain the optimal policy, and work as described below.

For every indexing agent, an integer value t_i (the index) is associated with each arm i. The agent then acts as follows: Initially, it selects a few arms at random to assess the mean of good and bad states across the five arms. Then, it chooses its next action by looking at the reward obtained by its last choice. If the arm selected

in the last round is evaluated as being in the good state, the agent selects it again; otherwise, it chooses at random from the subset of arms i that have not been selected for at least t_i time steps, where t_i is the index for that arm. In other words, the index of an arm denotes how long an agent must wait, after the arm is detected in the bad state, until it starts considering it again.

The testbed simulated 5^{15} such index policies ($t_i \in [1, 15], \forall i$).[4] Results from the simulation are presented in Table 2.1. (All differences are significant at the 1% level according to pairwise non-parametric Wilcoxon tests.) Naturally, policies (iv)–(vi-i) are not candidates for direct comparison, because they "cheat" by having access to knowledge the remaining agents do not; nonetheless, they are good indicators for the performance of the GRUE agents.

Agent	$R = 30$	100	500	1500
Average Emotion	3.64	3.26	3.07	3.17
i. Random	1.91	2.03	1.94	2.37
ii. Reactive	1.29	2.98	2.83	2.45
iii. Learner	3.15	2.93	2.32	2.22
iv. All-seeing	8.75	8.73	8.83	8.72
v. Half-seeing	3.99	3.70	3.62	2.92
vi-i. Best-index	5.64	4.46	4.13	4.16
vi-ii. Avg-index	2.44	2.47	2.46	2.43

Table 2.1: Average payoff of agents in restless bandit domain: Agents (i)-(iii) have the same knowledge of the world as the emotion-based agent; agents (iv)-(vi-i) know the state transition function and/or the actual state of all arms, while agent (vi-ii) represents a randomly chosen policy from the space of (exponentially many) optimal policies. The performance of the emotion-based agent is the average across the different choices of thresholds for the computational operators. It consistently outperforms other techniques that have the same knowledge as itself.

[4]This brute force approach was chosen due to the fact that the optimal index policy cannot be computed without having knowledge of the means of all states and the transition probabilities between them, which the emotion-based agents did not know.

As can be seen in the table, the emotion-based agent consistently outperforms the random, learning, and reactive heuristics. It also comes close to the optimal index policy (which, is, however, not computable without a priori knowledge of the parameters of the bandit arms). Choosing a policy at random from within the exponentially large space of index policies ('Avg-index' in Table 2.1) performed poorly. This is because the set of index policies may contain a large number of obviously bad or nonsensical policies. The randomly-chosen policy's performance indicates, however, that the requirement for optimality is strong when it comes to index policies. In other words, it is not sufficient to deploy an agent with *some* index policy and hope that it will perform well—good index policies are very rare. In fact, only about 50 of the 5^{15} index policies outperformed the emotion-based agent for every choice of R.

Note also how the domain impacts the performance of some of these other agents. The learner agent in particular appears to achieve diminishing payoffs as the running time of the simulation (R) grows. This is counter-intuitive, as one would expect more payoff samples to lead to better performance. I hypothesize that the reason that the learner agent experiences diminishing returns is that the information it assumes to have learned is more likely to become outdated as time passes. For example, if it receives payoff 5 from arm 2 at time 1, it is reasonably safe to assume that arm 2 will offer it something around 5 at time 2. However, as time goes on, the probability that arm 2 has transitioned to another state grows. As this happens with all k arms, the learner algorithm's beliefs are even harder to maintain accurately. On the other hand, the performance of the emotion-based agent seems robust with respect to the time R allotted for learning and experimentation.

Next, the robustness of emotion-based agents with respect to the choice of thresholds θ_1 and θ_2 was tested. Across all choices of R, the variance in the payoff obtained among the six variants of the emotion-based agents (with different choices of θ_1 and θ_2) never exceeded 0.3. Table 2.2 shows the performance of six variants for $R = 1500$. As can be seen, all but one variant (6) still outperform the other heuristics (last column of Table 2.1).

Variant	1	2	3	4	5	6
Avg. payoff	3.51	3.79	3.01	3.76	3.23	2.36

Table 2.2: Average payoff of emotion variants for $R = 1500$: The variation in performance across the various choices of thresholds for the computational operators is very small.

Foraging

GRUE was next evaluated in a foraging domain consisting of a 10×10 square field. Random units of food are placed on squares of the board, such that their total number does not exceed 15. A single agent starts in location $(0, 0)$ and is allowed, in every time step, to either (a) walk in one of the four directions, spending 5 HP, (b) run, covering twice as much space, but spending 20 HP, (c) pick up food, if it stands on it, or (d) eat food, if it carries some. An agent may carry up to one unit of food at each time, and picking up a unit of food causes one more to randomly appear somewhere, such that the total remains 15. The field is also populated by a number of enemies (8) and a number of friends (8), which are indistinguishable to the agent. Standing on the same square as an enemy in the end of a round causes the agent to suffer damage of -100, while a friend provides 100 units of benefit. Both friends and

enemies will move in every round toward the agent, if it finds itself next to them. Eating food also gives the agent 200 HP. There are two seasons (winter and summer) in this environment, and the season changes stochastically with probability 0.02 after every round. The season modulates which parts of the grind food is more likely to appear in. The location of friends, enemies, and food is not uniform; they are all clustered in parts of the field based on the current season (e.g., in the winter there are more enemies in the top-left corner). Agents are assumed to perceive the area that is one step away from their current location, counting diagonal steps (i.e., the 3×3 square surrounding them). However, agents do not observe the season, do not know the transition probability for its change, and have no knowledge of the movement patterns of friends or enemies. (For this reason, the emotion-based agent's internal state-space representation, Σ, only distinguishes between the presence of food, as well as the presence of a friendly or hostile agent in the vicinity of the agent.)

The emotion-based agent had two goals: $g_1 =$"avoid danger" and $g_2 =$"replenish HP." The presence of food and the acts of picking it up and eating it were set to increase the degree of achievement of g_2, while the presence of another creature (friend or enemy) was initially expected to decrease the degree of g_1. However, after an interaction, and depending on its outcome (experiencing damage or benefit), the degree of achievement of g_1 would go down or up. Hope would ensue after a positive interaction and decrease the priority of g_1, while fear, caused upon suffering damage, would increase it. The net effect of hope and fear was, therefore, a tendency to avoid creatures right after being damaged by one (since now g_1 had a high priority), and not avoid them after being benefited (since now g_1 had a low priority).

This emotion-based agent was compared against a random policy, and three scripted heuristic policies of the form: "walk about at random, pick up and eat food when you see it, and move in the opposite direction of other creatures." The three heuristic policies differed in whether the agent would walk or run away from enemies, and whether it would consider interacting with a creature to see if it is a friend or not. The agents were allowed 500 moves on the field and then their HP was recorded; for all agents the experiment was repeated 100 times.

The optimal policy in this problem would be the solution to a very complex POMDP with over 8 million states. To get a better sense of the performance of the system, however, features of the problem were abstracted and the number of states was reduced by assuming certain independencies. Thus, only the location of nearby creatures (up, down, left, right, center, none), the location of food, whether food is being carried and whether the very last interaction was with a friend or enemy were accounted for, resulting in an MDP with a total number of 144 states. (The abstracted problem is an MDP since unobserved variables, like the location of unseen creatures, are assumed not to matter.) In Figure 2.2 the performance of the various agents is presented. (Differences are significant at the 1% level.)

As can be seen the emotion-based agent outperforms the most obvious heuristics and comes very close to the MDP which was formulated on the reduced (abstracted) problem. Yet although the emotion-based agent underperforms the MDP, the latter requires more computational resources, and also requires some knowledge about things that the emotion-based agent does not have. For example, the MDP needs to know the transition probabilities between states, which in turn requires making

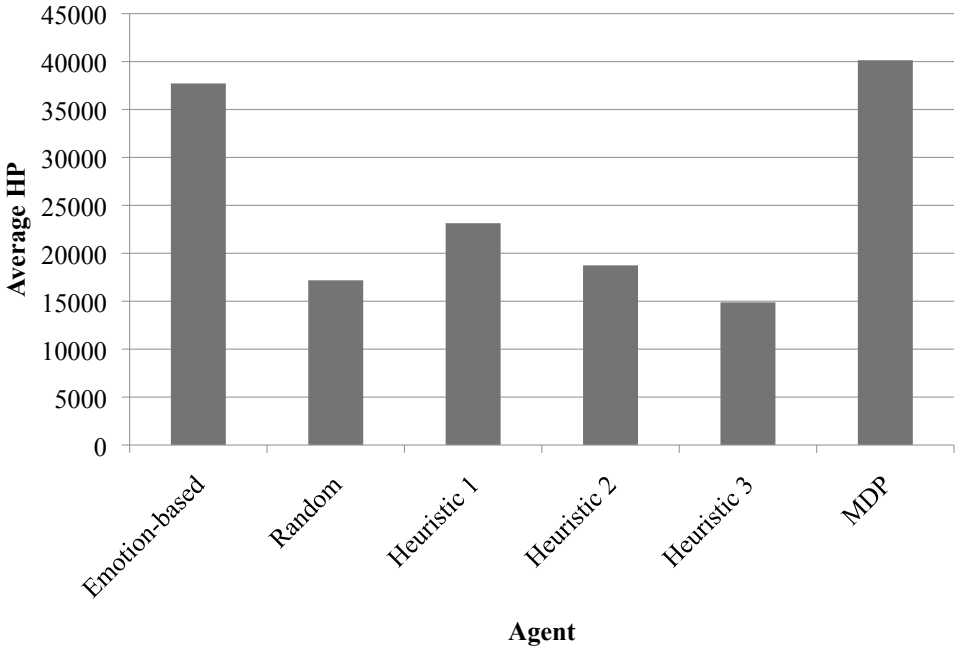

Figure 2.2: Average 500-round HP in foraging environment: The emotion-based agent performs close to the optimal policy for the simplified MDP representation of the domain, yet it requires much less computation and no human insight.

assumptions of the behavior of friends and enemies in the environment. Furthermore, abstracting away the original problem down to a more manageable MDP makes use human intuition and effort, which the emotion-based agent do not require.

2.3 Related Work

In this section, three types of approaches to complexity in AI are presented. One type aims to address issues of scale, for instance by reducing the number of states an algorithm might have to consider. A second sacrifices optimality guarantees. And a third type exploits independencies in the domain. The unifying theme of all three approaches is to exploit *structure* to reduce the size of a representation or the computational requirements of an algorithm.

Abstractions and Discretization

As AI tackled problems of increasing size, it became obvious that the domain could not be fully represented, and that it is impossible to reason through every aspect of it or every contingency that might occur. Sometimes this happens because of the sheer scale of the domain. For instance, to provide an optimal algorithm for the rather simple game of Texas Hold'em Poker one would have to consider all the possible combinations of cards on the table and in other players' hands. Other times the state space is continuous, and states cannot be considered individually. In the game of billiards, for instance, the state space (i.e., the location of the spheres) is continuous. The action space, represented as the position, angle and velocity of a strike, is also continuous.

One insight necessary for handling problems of scale is that not every state or every detail probably matters a lot by itself. A large number of states or actions might be "equivalent" for the purposes of making a decision and might afford similar patterns of thought. For example, in poker one might plausibly assume that all hands not containing a high value or an interesting combination ("weak hands") can be played similarly. Also, for player i it might not matter a lot if opponent j or opponent k have a particular hand for the purposes of deciding whether to call or fold, as long as one of them has it. Essentially the original game of poker might be suffering from large scale, but the decision-making algorithm operates on a reduced version of it [45, 117]. Similarly in billiards an algorithm might be more efficient by assuming that several symmetrical positions on the table are equivalent [5]. Such useful simplifications *abstract* away from the specific details of the problem and allow an algorithm to work on a higher level, presumably in a much reduced state and action space.

Heuristics

Heuristics are another set of techniques to deal with complexity, closely related to abstractions. Algorithms that use a "quick and dirty" approach to decision-making and problem-solving can be typically thought of as making use of a heuristic. These heuristics avoid extensively deliberating about a choice in order to identify the optimal decision, and use simpler, local metrics to decide. For instance, *greedy* heuristics make a myopic decision based on what seems best "right now," overlooking that fact that such a local decision might not necessarily lead to the best possible solution over

time. Other heuristics might use randomization (e.g., take an action uniformly at random) or make a choice that is contingent upon what is happening in the world (e.g., 'copy what your opponent did last round'). Sometimes heuristics are part of bigger algorithms and—assuming that the heuristic satisfies certain desirable properties, such as *admissibility*—these algorithms may in fact guarantee optimality (such as A* citeastar) or provide guarantees as to how far away from optimal its performance will be.

Independence and Graphical Structures

Another way of exploiting structure to simplify the representation of an environment (and therefore reduce its complexity) is to take advantage of possible independencies in the state space. Imagine, for example, an algorithm that monitors the operation of a car manufacturing facility. Several errors might occur along the production line (such as misplaced or misshaped parts), in the warehouse, or the electrical and ventilation systems, among others. The system utilizes a number of sensors to identify errors, diagnose their source and, if possible, correct them. In this setting, almost every part of the system is interconnected, such that, for instance, a local power failure might lead to a robot malfunctioning in the production line, and thus a broken car needing to be loaded off the truck later on. Yet the algorithm can be greatly simplified by acknowledging that most interactions are localized. In other words, each sensor reading is significantly correlated with only a small number of direct causes. Other than these explicit correlations, other quantities are said to be *independent*. Such independencies make reasoning and decision-making easier: observing a partic-

ular sensor reading the algorithm only needs to reason about the quantities that are correlated with it, and hence with the set of actions relevant to these diagnoses. In our car manufacturing example, for instance, misshaped car parts are independent of the warehouse loading/unloading schedules. Therefore, when such parts are observed in a car, the algorithm should only probe the state of the manufacturing robots or the quality of the basic material involved.

Independencies like that are captured predominantly using graphical data structures. Such structures were introduced in the late 1980s and became known as Bayesian Networks (BNs [102]). Bayesian Networks encode joint probability distributions in efficient ways that take advantage of independence relationships. Bayesian Networks and their extensions (Influence Diagrams or IDs, and Multi-Agent Influence Diagrams or MAIDs) are useful in decision-making. In particular, MAIDs have been used to simplify Nash equilibrium computation in certain domains [71], sometimes resulting in exponential time savings. Other extensions of Bayesian Networks have also been used for decision-making, among them object-oriented Bayesian Networks [69] and relational Bayesian Networks [59], Hidden Markov Models (HMMs, [105]) and Hierarchical Markov Models [39].

Other graphical representations aim at reducing complexity by taking advantage of independencies in the strategy space. *Graphical games* [65] represent players in a game as nodes on a graph, and add a directed arc between two nodes i and j when the actions of player i can influence the payoff of player j. This representation seeks to exploit cases in which the payoffs of most players only depend on the strategies of a few other players, and therefore are independent of a large part of the strategy space.

Another representation is that of *action-graph games* [61], which capture another sort of useful independence. In particular, in many games the payoff of a player depends on whether a particular action has been taken by somebody, but not necessarily on *who* took that action. In both formalisms it has been formally shown that the representation is greatly reduced when these independencies abound in the structure of the game.

Additional Approaches

Several other techniques have been used to solve for good decisions and identify well-performing strategies, that do not neatly fit in any of the above categories. With respect to game-theoretic approaches, the field of *algorithmic game theory* has attempted to develop techniques that simplify equilibrium computation. The algorithms developed typically perform well on particular representations (e.g., the continuation method on MAIDs and graphical games [14], or the sequencing method in sequence-form games [70]), or they might assume particular classes of games.

Decision-making also often involves reasoning about unobservable or partially-observable quantities. For instance, *games of awareness* [107] try to explicitly reason about situations in which agents might not be fully aware of the options other possess (their strategy space). When certain Markovian assumptions hold in the domain, as in the case of MDPs and POMDPs (see Section 1.2), tools and algorithms such as *Interactive Partially Observable Markov Decision Processes* (iPOMDPs [49]) attempt to relax the assumptions of common knowledge and explicitly represent nested belief hierarchies, refining equilibrium solutions to reflect this uncertainty.

Other emotion-based analogues for decision-making

Prior work has also considered the use of computational analogues of emotion in agent decision-making. In that space, we can distinguish between two types of approaches. In the first type, computational analogues of emotion are incorporated into agent design to give them human-like traits. This is especially useful in designing virtual agents that will interact with people. For example, a virtual agent that converses with a person may appear more natural if it can compute and exhibit appropriate emotion expressions, or if its language can be meaningfully influenced by the affective content of the conversation. Such computational analogues of emotion have been used to create sophisticated virtual agents. Some of these agents are accurate enough to be used for the training of psychotherapists, since they can exhibit emotional responses similar to those of actual patients [66]. Others interact with people in an affect-modulated way to built "rapport," making the interaction seem more natural and thereby influencing people's perceptions and behavior [51, 29, 30].

The second type of work has incorporated computational analogues of emotion into agents with the aim of increasing their performance, not of producing more natural human-computer interaction [111, 110, 112]. GRUE differs from these models in that it uses computational analogues of emotion that are consistent with established and well-studied appraisal-based psychological theories. Scheutz [111], for example, proposes a reactive decision-making algorithm that maintains a "mood" quantity for each action the agent can take. When the agent considers its various candidate actions, this mood measure is used to skew their expected utilities. Hence, if negative mood is associated with an action, the expected utility of that action is perceived

to be smaller. Although such techniques lead to performance increases in certain domains, they cannot be characterized as properly "affective," since their heuristic notions of affect are not generated by a consistent account of the emotions. Due to this, they suffer from two issues in practice: First, undifferentiated notions of affect like that are inherently limited in the way they may assist an agent. Fear and anger, for example, are both negative emotions but have important differences and assist a person in very different situations and in very different ways. Treating them both flatly as a "negative mood" misunderstands and misapplies them. Second, unless a theory of emotion is used (e.g., cognitive appraisal), the process of maintaining and updating the emotional state of the agent must rely on heuristics, and these are typically highly domain-specific and difficult to generalize. By being grounded on established theories of emotion, GRUE is able to better distinguish between the various emotions and has a domain-independent appraisal-based method to update the agent's emotional state.

2.4 Discussion

GRUE uses quantitative notions of certain human emotions to make decisions in complex environments. In two complex domains, in which the optimal solution is either very hard or intractable to compute, it was shown that agents following GRUE performed well and required very limited computational resources. Furthermore, the emotion-like operators in GRUE were formalized as domain-independent operators. As such, they hold the potential to be useful in a variety of other novel environments.

This method is complementary to other decision-making approaches presented

earlier in this chapter, and shares many similarities with them. As already argued, the emotion-like operators can be thought of as reactive heuristics, fine-tuned by evolution to respond to a broad range of similar situations [128]. In this way, emotions also behave as a form of abstraction, since they explicitly overlook the vast richness and subtlety of the world to make a decision based on only a small number of features (e.g., in the case of fear, only based on whether a source of danger has been detected). Emotions also detect independencies, focusing people's thinking and reasoning on what is important "right now" [55, 25] and treating everything else as not directly relevant. Similarly, the operators used by GRUE respond to stimuli by changing the priorities of the agents' goals, thus focusing the agent's decision-making toward achieving goals that currently have a high priority. Hence emotion-like operators form a particular class of heuristics that intelligently *bundle* aspects of abstraction techniques and independence-exploiting tools. The advantage of these bundles stems from the fact that they have been fine-tuned over millions of years of evolution to perform well in practice in frequently encountered situations throughout the course of evolution. As a result, to the extent that agents operate within environments that share many structural similarities with those ancestral environments, we would expect these emotion-like operators to be helpful in guiding them to good decisions.

The use of the word "emotion" in GRUE must be understood as detached from the biological and psychological complexities of emotions in people and living organisms in general. Although cognitive appraisal theories in psychology [100, 73, 99] have taken an information-processing approach to analyzing emotions, the biological substrate and the psychological influences of emotions are far too complex to account for in a

simple model. Moreover, certain functions of affect and insights were brought into agent design over to the extent that they confer a performance advantage or reduce computational cost. In that spirit, the goal in this thesis has *not* been to replicate human emotional reactions with high fidelity, but to adapt basic notions of emotion-based decision-making to the practical needs of AI agents.

In that sense, the emotion-based computational operators bring those behavioral responses, crafted and refined by evolution, into agent design, to produce a flexible agent architecture. In this architecture, the various emotions represent appropriate responses to certain classes of situations. For example, the operator of *boredom* is a good response in environments where great learning potential exists and the agent risks spending too much time and effort in one particular subspace of the domain. Boredom causes a perturbation of the goal priorities to move the agent out of a stable (but potentially suboptimal) situation and induce a higher degree of exploration. Other emotions are useful in other kinds of situations and bring about different adaptations. As a whole, the method is similar to other portfolio approaches like IBM's Watson [38] or the Netflix prize winner [12], in that emotions can be seen as modules in an architecture, with each module working well in a particular situation and for a specific problem, and all of them (as a whole) exhibiting high performance.

It must also be noted that the simulations described to evaluate GRUE served to examine very complex domains, for which optimal solutions are either not tractable or very hard to compute. On the one hand, this illustrates the potential value of emotion-based decision-making operators. On the other hand, the absence of more sophisticated algorithms necessitated comparisons between the emotion-based agents

and relatively simple heuristics, as no better domain-independent solution approaches exist that can be used without extensive periods of learning.

Future Work

Despite being justified in using simple heuristics to compare GRUE against, the simulations say little about domains for which optimal solutions are currently within our reach. In those cases, the agent designer might have to select between using an emotion-inspired approach or an explicit optimization tool. The former would allow building the agent faster, but it would not be able to provide any optimality guarantees. In those cases I would recommend experimenting with emotion-inspired agents (perhaps in simulation) and establishing how good their performance is in practice, before full deploying them.

Possible future extensions of GRUE include exploring how this methodology lends itself to generic domains and how it compares with currently optimal solutions. One of the avenues worth exploring is to compare emotion-based agents with "anytime" algorithms, and therefore assess the computational cost (in time and resources) for these algorithms to catch up with an emotion-based decision-maker which operates "out of the box." Quantifying in this way the limits of this emotion-inspired approach, one will have a better sense of when an emotion-based methodology is appropriate or not.

Another possible extension is to relax the assumption of myopic choice. Clearly, people do not simply rely on short-term, instinctive or emotional assessments to reach good decisions, they frequently deliberate about the future. Most of the time they

follow a combination of short- and long-term decision-making approaches. It is an open question whether and how deliberative algorithms can be extended to use short-term or myopic assessments as submodules, and whether myopic approaches can be extended to take into account limited lookahead or perform computations that will make them more deliberative, accurate and precise.

Chapter 3

Decisions in Groups: Affective signaling

Complex environments rarely involve a single agent reasoning and acting alone. Others—humans or computers—are part of the world, and their actions are critical for the agent's decision-making. The presence of other agents raises two new issues. First, for an agent to respond successfully to what others might be planning, it needs to *predict* their actions as accurately as possible. In collaborative domains, in which the goals of everyone are aligned, having a good predictive model of others can help minimize mis-coordination and improve efficiency; and in competitive domains, in which agents are self-interested, knowing how others will act allows one to best-respond to their potentially adversarial plans. Second, an agent must be able to *interpret* others' actions after they occur, as doing so helps the agent better identify the strategies others are following, their preferences (utility function) or goals, and any other unobservable characteristics information they might possess.

This chapter presents an affective mechanism for these two problems of prediction and inference in multi-agent settings with high uncertainty. The central idea behind this mechanism is that observing other agents' actions is not sufficient for disambiguating and interpreting their behavior. There might be a multitude of interpretations consistent with the same action of an agent. To reduce ambiguity, agents are designed to produce and transmit an emotion-like signal that reflects their "reaction" to what is happening in the environment. For instance, if they perceive that something undesirable has just happened to them, they emit a "distress" signal. This signal is ambiguous on its own as well (distressed about what?), but in *conjunction* with the observed action and the context in which both are observed it serves to reduce ambiguity.

Agents following the mechanism produce this signal *unintentionally*, in the sense that it is not strategically and explicitly computed—its production and transmission is happening automatically and without the agent having to reason about the effect of the signal on others' inferences, perceptions and future actions.

Think about an example from natural language, and consider a parallel. The words in a sentence are like explicit direct actions, and intonation is akin to a supplementary signal. Imagine then reading the phrase "She did a good job." This phrase can have multiple interpretations depending on intonation. One may utter this sentence as a statement of fact. One may also emphasize the person who performed well ("*She* did a good job."). Furthermore, one may add surprise, derision or commendation. Finally, one might place emphasis on how good her performance was ("She did a *good* job!"). Words alone are not enough to disambiguate, and intonation

without words carries little information, but both of them together help make the signal explicit.[1]

Section 3.1 begins with a brief overview of inference and communication in multi-agent systems from a game-theoretic standpoint. I discuss issues pertaining to unobservable characteristics of agents, private information, Bayesian games, individual private priors, and the difficulty of making inferences when the assumption of a fully-known small world is relaxed. Previous approaches to communication and inference are also presented. Section 3.2 moves the discussion from game theory to prevailing views in evolutionary psychology and cognitive science about the communicative functions of the emotions. In these accounts affect is seen as a concise yet powerful communication tool, revealing a person's state but also providing useful information for groups to coordinate their activities.

According to several models of human emotion, affect is generated by means of a set of *appraisal variables*. These variables capture how the agent reacts to events in the world. "Desirability" is one such variable, measuring how conducive an observation is to the person's goals. Another appraisal variable is "blameworthiness," which captures the extent to which another person is to be held responsible for his actions (it was intended and premeditated). The mechanism uses three such appraisal variables by which agents can send emotion-like signals to others around them. Moreover, these variables are easy to compute, making the signal generation practically costless. Agents receiving the signal use Bayes' rule to infer the unobservable characteristics of

[1]Of course, emotions such as surprise and derision are partly unintentionally communicated, but sometimes intentionally. In this work I assume a strictly unintentional signal, deferring the handling of strategic signaling to future work.

the sender, by conditioning a prior over such characteristics on any actions the have observed and any signals they have received.

Section 3.3 evaluates the mechanism in two domains: an iterative "social dilemma" (an extension of the finitely repeated prisoner's dilemma) and a task domain modeled as a DEC-POMDP with private utilities and private state information. The first domain takes a population of agents playing a very simple game and adds just a single unobservable characteristic. In particular, both cooperative and self-interested types of agents are allowed to exist in the population. This is sufficient to increase the complexity of the game, raising problems which the proposed signaling mechanism addresses. In particular, agents are randomly paired to play a game, during which measurements of how quickly and accurately they are able to infer their partner's type are obtained. It is demonstrated that when agents transmit affective signals during their interactions, they can infer their partner's type more quickly and more accurately, which results in improved performance. The second domain considers agents in an environment in which there are opportunities for collaboration, and measure whether affective signals are an *efficient* way of transmitting relevant information. The simulation compares between populations of agents who emit no signals, agents who communicate their full internal state, and agents that send this "low-bandwidth" affective signal the mechanism generates. The results illustrate that affective signals lead to similar improvements in agent coordination as full-state transmission, but with much less costly communication.

Throughout these three sections, signals are assumed to be truthful. Agents do not fabricate them to strategically mislead others. Section 3.4 argues that this is a rea-

sonable assumption. It presents a computational conjecture that in complex domains manipulability of a signal might be computationally hard, particularly for repeated interactions. The chapter concludes with a discussion and possible extensions of this work.

3.1 Inference and Communication in Multi-Agent Systems

In the language of game theory we can distinguish between games of *perfect*, *imperfect* and *incomplete* information. In perfect information games everything about the game is assumed to be common knowledge, such as every agent's payoff function, the actions available to it, and the structure of the game (i.e., which agent goes first, who moves after every action sequence, etc.). Moreover, at every point during a game of perfect information, all agents know exactly the actions everybody else has taken so far. When perfect information games are represented in the extensive form (see Section 1.2.2) all information sets are singular nodes and the optimal strategy (Nash equilibrium) can be identified simply by using backwards induction: start at the leaf nodes of the tree and go upwards, selecting at every node the action that maximizes the utility of the agent whose turn it is there, until the root of the tree is reached; the strategy profile consisting of the decisions selected at every node is a Nash equilibrium for the game as a whole.

In imperfect information games some agents may have information sets consisting of two or more nodes, because either the world is stochastic (in which case an addi-

tional player, "Nature," is assumed to be making probabilistic choices at its nodes along the tree), or because these agents do not observe an action taken before their turn (including perhaps some of their own past actions, if the game does not possess the property of "perfect recall"). Nash equilibria in games of imperfect information cannot in general be identified by backward induction, and more complex algorithms are needed.

In the final category of games (incomplete information), the unobservable information is structural. Players might be uncertain of each other's payoff functions, or even the actions/strategies available to them. Furthermore, players need to reason about the *beliefs* of others. Suppose, for example, that player i believes player j's utility function is u_j or u'_j with equal probability 0.5. However, player i might acknowledge the fact that j knows with certainty which function her true utility is. Player i might even believe that player $k \neq j$ disagrees on the probabilities assigned to u_j and u'_j. To model situations like that the concept of a "type" is typically invoked. The type of an agent encapsulates its utility, its action set, its beliefs about the structure of the game, and its beliefs about other agents' payoffs, action sets, etc. In a sense, the type of an agent fully characterizes that agent's entire knowledge and beliefs, and remains fixed during the course of a game.

Games of incomplete information are further distinguished into games with a common prior and games with individual priors. In the former, there is a commonly-known probability distribution that assigns types to players at the beginning of the game. Games of incomplete information with a common prior can be represented as games of imperfect information, by having Nature assign types to players in the be-

ginning of the game, such that each player i is only aware of her own type assignment. Games of incomplete information with individual priors cannot be represented as a single game of imperfect information, as the agents would disagree on the probabilities that Nature would use to assign types at the beginning of the game.

When uncertainty exists, agents are assumed to take actions that maximize their expected utility. In a multi-agent setting, however, computing expected utilities is significantly harder, because agents' payoffs depend on each other's actions, and are cannot be sure what strategies others are following. In other words, there exists *strategic uncertainty*. Strategic uncertainty has several sources. In games of perfect or imperfect information all agents will agree on the set of Nash equilibria of the game, but this set might be large. Therefore agents cannot be sure that everyone will choose the same equilibrium in this set to implement. As a result, the strategies followed by all the agents in practice might not be in equilibrium. In games of incomplete information this problem is exacerbated by not knowing others' types (e.g., utility functions). In those settings, the equilibrium solution concept must be amended to reflect this. In a Bayes-Nash equilibrium, agents best-respond to the strategies of all other agents of all types, each weighed by the probability of that type, according to the common prior (if that prior is assumed to exist) or that agent's individual prior.[2]

3.1.1 Inference and Implicit Signaling

To perform well in the world agents may benefit from taking steps to reduce their uncertainty. In collaborative domains, they might wish to reduce not just their own,

[2]In games with individual priors agents will fundamentally disagree on the set of Bayes-Nash equilibria of the game, making the problem of strategic uncertainty even harder.

but also others' uncertainty in order to improve coordination. In competitive domains, on the other hand, they might wish to increase others' uncertainty to confuse and mislead them. An agent can reduce its uncertainty in various ways: First, it can passively observe other agents' actions as the game unfolds and thus infer their type. In games of imperfect information this is somewhat similar to *forward induction*, by which agent i uses its observations of agent j's actions to infer j's future behavior down the game tree, i.e., the equilibrium strategy j is implementing [44]. Second, the agent can actively take actions (exploration) to reveal more about other agents' type or private information. For instance, in the game of poker, a player can infer whether her opponent has a good hand by observing her bidding behavior, or he can adjust his own bidding behavior (bluffing) to actively test whether she has a good hand or not. Note that in the second case the agent is "speaking by doing," i.e., the communication act is also a game action that is part of the agent's strategy—a bluff in poker is also a betting action.

Inferences in game-theoretic agent designs are typically made using Bayes' rule. Imagine a game with two players. Player 1 knows its own type ($t_1 \in T_1$) but its beliefs over player 2's type are given by a distribution, in which each $t_2^j \in T_2$ has probability p_j (with $\sum_{j=1}^{|T_2|} p_j = 1$), where T_i denotes the typeset for player i. Imagine also that both players implement a single Bayes-Nash equilibrium σ, and action a of player 2 has probability π_j to be taken by type t_2^j of player 2, for all $t_2^j \in T_2$. After observing a, player 1 will update his belief over 2's type by conditioning his prior on that observation. Hence, player 2 will be of type t_2^j with probability proportional to $p_j \times \pi_j$. In general, after seeing action stream (vector) $\mathbf{a}_j$ by player j, player i will

update his prior $\Pr[t_j]$ over j's type according to:

$$\Pr[t_j|\mathbf{a}_j] = \frac{\Pr[t_j] \times \Pr[\mathbf{a}_j|t_j]}{\Pr[\mathbf{a}_j]} = \frac{\Pr[t_j] \times \Pr[\mathbf{a}_j|t_j]}{\sum_{t'_j} \Pr[\mathbf{a}_j|t'_j]}$$

In games with more than two players the above quantities easily generalize: after observing actions $\mathbf{a}_{-i}$ by all other players, agent i updates his belief about their types t_{-i} by using Bayes' rule:

$$\Pr[t_{-i}|\mathbf{a}_{-i}] = \frac{\Pr[t_{-i}] \times \Pr[\mathbf{a}_j|t_{-i}]}{\Pr[\mathbf{a}_{-i}]} = \frac{\Pr[t_{-i}] \times \Pr[\mathbf{a}_{-i}|t_{-i}]}{\sum_{t'_{-i}} \Pr[\mathbf{a}_{-i}|t'_{-i}]}$$

How hard are these these inferences? In simple settings they are often computationally trivial, but in complex environments inferences become significantly harder. This is evident in the above formula. In complex domains three factors make inference hard: First, the typeset T_{-i} might be very large. When that happens, representing the distribution might be onerous, and computing the denominator sum (over all $t_{-i} \in T_{-i}$) becomes computationally expensive. Second, actions might not be very informative. This is because action stream $\mathbf{a}_{-i}$ might be indicative of many types in T_{-i}, that is, $\Pr[\mathbf{a}_{-i}|t_{-i}]$ is similar for many $t_{-i} \in T_{-i}$. As a result, observing $\mathbf{a}_{-i}$ does little to disambiguate between all those different types. In poker, for instance, 'raise' might be the action associated with hundreds of different hands the opponent might have.

If other agents in a game are performing inferences like that, an agent is essentially *signaling* its type (or any private information) through its actions, either passively or actively. In passive signaling the agent is merely choosing a strategy to maximize its expected utility and by doing so it essentially reveals (partly or fully) its type to other agents. An example of such signaling is the famous separating equilibrium [84].

In games with a separating equilibrium, agents of different types in equilibrium take different actions, essentially revealing their type. In the most well-known example of this type of equilibrium there are two players, a 'worker' and an 'employer.' The worker may belong to one of two types: 'smart' or 'not smart,' but employers cannot observe a worker's true type. In the beginning of the game the worker can choose to 'receive education' or 'skip education;' after that, the employer chooses to 'hire' or 'not hire' the worker. Under the appropriate values for their payoff functions, smart workers will optimally choose to receive education (because for them it is easy and thus of low cost) and non-smart workers will choose to skip education (because for them it would be hard and thus costly). After that, employers will simply hire workers who have chosen to receive an education. In this way the workers' actions act as a signal that separates the two types in equilibrium.

Not all cases are that easy, however. In many games there is no separating equilibrium, so the uncertainty about others' types remains after their actions are observed. Also, agents often wish to actively mask their type, throwing red herrings to confuse others. This is exacerbated in the case of *long-running* interactions with multiple (or infinite) rounds. Take the simple example of an online seller who can be either 'benevolent' or 'malevolent.' Benevolent sellers always ship the goods to their customers after receiving payments—their utility function includes a strongly negative "guilt component" if they walk out on their agreements. Malevolent sellers, on the other hand, have no such component and wish to maximize their net monetary gain. In this setting it might be advantageous for malevolent sellers to act in the same way as benevolent ones, building themselves a reputation for buyers to trust them. After

some time they might choose to "milk their reputation" and defect on a buyer, but until then inferring their type by observing their behavior is virtually impossible.

3.1.2 Explicit Signaling

In the discussion above we considered implicit signaling, in which the actions taken by an agent also serve as the communication to other agents. But agents may also signal information to others explicitly, using *communicative acts*. These actions may be part of the agent's strategy, but they do not directly influence its payoff. In the online sellers example, a benevolent seller may include "previous customer testimonials" to persuade buyers to trust him. This communicative action does not influence his payoff upon a purchase. It is essentially "doing by speaking." In the field of linguistics, one of the most comprehensive theories of communicative acts is the *speech act theory* [113], according to which communicative acts can be analyzed on three levels: a locutionary act, the performance of an utterance; an illocutionary act, its real, intended meaning; and in certain cases a further perlocutionary act, its actual effect, such as persuading, convincing, scaring, enlightening, inspiring, or otherwise getting someone to do or realize something, whether intended or not.

Explicit signaling is a helpful way to deal with the Bayesian inference problems described in the previous subsection. When an agent's actions are not very discriminative of its type, an explicit communicative act might direct other towards the correct type inference. In games with multiple equilibria, also, such communicative acts may help guide agents towards a "focal point," essentially adding coordination in the equilibrium selection problem. For example, in the famous "battle of the sexes"

game, there are two Nash equilibria in pure strategies that are symmetric and their only difference is that player 1 prefers the first equilibrium while player 2 prefers the second. Both players are better off coordinating but they disagree on which equilibrium they should implement. Player 1 in that instance may use a communicative utterance such as "I will always play the strategy corresponding to my preferred equilibrium." Such an utterance is strictly non-rational: if player 2 selects the strategy corresponding to her preferred equilibrium, player 1 will be strictly better off ignoring his public proclamation. However, if player 2 believes he will "stick to his word," she will rationally help him implement his desired equilibrium, which has now become the focal point.

As another example of explicit communicative acts, "signaling games" [23] are a particular class of 2-player games, in which an agent is trying to infer its partner's unobservable type by interpreting message it is sending. In POMDPs, also, communication has been used for both planning [108] and learning [126]. In the former case, communication is used to share information within a group of agents to improve coordination. In the latter, a group of agents is trying to explore the state-action space of a POMDP in order to learn the optimal policy, and communication among the agents allows them to decentralize and speed up the learning process.

Explicit signals can, however, only be trusted in collaborative games, in which all agents share the same utility function, so there is no incentive for anyone to lie about their type or any private information they possess. In contrast, because such explicit signals do not impact the signaling agent's utility, in competitive domains the agent might costlessly lie, and the signal is therefore undependable ("cheap talk").

The field of *mechanism design* addresses these situations. A mechanism is essentially a game whose very structure and payoffs aim at guiding agents' behavior towards desirable goals, such as efficiency (the outcome maximizes the sum of everyone's utilities) or truthfulness (the agents do not have an incentive to lie in signaling their types). Mechanisms have a variety of forms (e.g., auctions) and typically achieve the truthfulness property by *aligning* agents' incentives and causing agents to internalize the effect of their signals. In particular, agents are asked to pay a cost when they participate in the mechanism. This cost may depend on the signals communicated by every agent, as well as the outcome selected by the mechanism. Its functional form guarantees that no agent will improve its utility by being untruthful (see Ausubel et al. [7], as well as Jackson [58], and references therein).

Truthfulness is not the only issue with communicative acts. Communicating information may be costly, both in terms of how often agents communicate and the size of the messages exchanged. Communication protocols therefore need to reason about what to communicate and when, as transmitting all possible information constantly may be costly enough to offset the benefits of communication. Probabilistic Recipe Trees (PRTs) [64] have addressed this problem by using a decision-theoretic approach. In particular, PRTs reason whether the expected benefit of a communicative act outweigh its cost given the context and the beliefs of the agent.

3.2 Affective Signaling

To address the challenges of inferring others' unobservable characteristics, an affective signaling mechanism, inspired by the communicative functions of human emo-

tions, is deployed. People possess the ability to communicate, using facial expressions and other non-verbal cues (prosody, head movements, posture), a variety of attitudes toward what is happening in the world during their interactions with each other. For example, people may convey their satisfaction with a smile, or their regret with an expression of shame. In this work, a signaling mechanism is designed that has three important properties: (a) it is *generic*, not domain-dependent; (b) it is automatic, in the sense that generating the signal is not part of the agent's strategy; and (c) it is shown to improve the *performance* of agents, by enabling more accurate inferences earlier in the game.

The signaling mechanism is evaluated in two domains. In the first, populations of agents of varying types engage in multiple-round social dilemmas. Agents differ in whether their payoff structure promotes cooperation or defection, and also in the strategies they follow during the game. The generic, affective signaling mechanism presented here enables agents to better distinguish between potential cooperators or defectors, and the population of cooperators becomes more robust. These effects persist even if agents do not have accurate or common beliefs about the distribution of types in the environment. In the second domain, the case of two agents acting together in a stochastic environment represented as a DEC-POMDP is examined. If agents communicate their private state, they can significantly avoid conflicts and they can successfully assist each other when needed. More importantly, it is demonstrated that merely communicating with affective signals is equally effective as communicating one's entire private state information, despite the fact that affective signals carry significantly less information. This suggests that affective signals are capable

of conveying what is essential to good decision-making in collaborative domains.

These results carry significant implications for the design of agents in uncertain, complex environments. The findings suggest that the communication of simple affect-like attitudes about what is happening during an interaction carries the potential to make inferences quicker and improve performance. Furthermore, they indicate that the communicative aspect of human emotions, refined by millions of years of evolution of decision-making animals living in a complex real world, might provide a robust guideline about what to communicate and when.

Finally, throughout this chapter it has implicitly been assumed that these affective signals are always truthful. Yet agents might lie about their attitudes, in much the same way people sometimes fabricate their emotion expressions. A computational argument is presented in Section 3.4 on why such fabrication might be difficult in practice in general domains, and hence why being truthful by default might be beneficial.

3.2.1 Affective Generic Signals

Research by psychologists has shown that emotions serve two beneficial functions: a *cognitive* one, whereby emotions influence one's thinking and information processing, and a *communicative* one, which modulates what one's expressions communicate to others [4, 11, 128, 55, 25, 98]. Prior work has leveraged the cognitive functions of emotions to guide agents decision-making in complex environments [3].

This chapter focuses on the communicative functions of emotions. People usually convey emotions by facial expressions, prosody, language, gaze, posture, and

head position. Evolutionary psychologists have studied the usefulness of expressing one's emotions, and have been argued, for example, that expressing fear alerts others quickly to the presence of danger and that smiling is a relationship-building signal [109, 86, 123, 96].

Several computational models of emotion have been proposed in prior work to account for the emotions generated by humans in different situations [50, 133, 118]. Most of these models build on cognitive appraisal theories in psychology [73, 41]. These theories describe emotions as the result of an *appraisal* of a situation with respect to the person's goals and desires. For instance, fear is associated with an appraisal of a stimulus as a threat to an important goal, such as survival.

A signaling mechanism has been designed for communication within groups of agents by borrowing features from these computational models of emotion. Appraisal is mediated by a set of *appraisal variables* which define and quantify the impact of an observation of the state of the world on an person's goals. Three such appraisal variables are being used: *desirability* denotes how good the outcome is in comparison to the agent's expectations; *control* captures the ability of the agent to unilaterally secure itself a good outcome in the future irrespective of what others might do; and *self-blame* reflects the agent's regret about the outcome of an interaction. Of these variables, 'control' and 'self-blame' are assumed to be binary, and 'desirability' might be positive, neutral, or negative.

It is important to emphasize that although the affective signaling mechanism makes use of these appraisal variables, agents that implement it do not manifest emotion expressions identical to those of people. In other words, although useful

concepts from the communicative functions of human emotions are borrowed, the mechanism does not aim to be a high-fidelity imitation or model of how people behave.

	High Control	*Low Control*
+ Desirability	Joy	
0 Desirability	Sadness	
– Desirability (w/o Self-Blame)	Anger	Sadness
– Desirability (w/Self-Blame)	Shame	

Table 3.1: Signals expressed based on levels of desirability, control, and self-blame

The signal that an agent communicates depends on the three appraisal variables as shown in Table 3.1. The signal predominantly conveys the 'desirability' of an observed event. The other two variables, 'self-blame' and 'control,' only influence the agent's communicated signal in the case of a negatively-appraised event. In particular, if the agent obtains a higher-than-expected outcome, given its current beliefs, it communicates joy. If it obtains more or less what it expected, it communicates a neutral emotion. And if it obtains less than expected, it communicates anger if it has a safe option (high control), sadness if it does not have such a safe option (low control), and shame if it considers itself the cause of its low payoff and regrets the action it has taken. It is assumed that the signal is generated automatically and non-strategically through these appraisal variables and is therefore truthful. If agents have the ability to reason strategically about their signal, they might have an incentive to fabricate it. Additional future work will be required for the analysis and handling of such instances.

Recipients of these signals use them to infer unobservable characteristics of the sender. Some of these unobservable characteristics are fixed over time, while others might change during the course of an interaction. Agents' utility functions and the actions available to them are, for example, typically fixed. In the language of game theory, such information is often referred to as the agent's '*type*,' a term that is adopted here. Unobservable quantities that change over time might include the value of internal variables, whether an agent has taken a particular action in the past, or has obtained a certain observation. The term '*private state information*' will be used to refer to such changeable quantities.

The process of inferring the unobservable characteristics of agents is modeled using Bayes' rule. To infer others' type, for example, each agent possesses a prior that captures its beliefs about the distribution of types in the population. Agents might have the same common prior or distinct individual priors. An agent conditions its prior on observations of the other agent's actions. For example, after observing actions $\mathbf{a}_{-i}$ by all other agents, agent i infers the likelihood of their types t_{-i} by computing $\Pr[t_{-i}|\mathbf{a}_{-i}] \propto \Pr[\mathbf{a}_{-i}|t_{-i}]p_i(t_{-i})$, where p_i denotes agent i's prior. Signals are treated in a similar fashion, with each agent conditioning its prior on both the actions of its partner and the signals it has broadcast, hence computing $\Pr[t_{-i}|\mathbf{a}_{-i}, \mathbf{s}_{-i}] \propto \Pr[\mathbf{a}_{-i}, \mathbf{s}_{-i}|t_{-i}]p_i(t_{-i})$, where $\mathbf{s}_{-i}$ denotes the signals communicated by the other agents.

Inferring others' private state information works in a similar fashion, by conditioning a prior on observed actions and communicated signals. As state changes over time, however, it is usually more challenging to infer state than to infer type. If agents

possess knowledge of how others' private state information changes stochastically over time, for instance, they can use this knowledge to more accurately track its evolution across time.

To compute the value of the three appraisal variables, an agent compares its expectations, given its beliefs, with the actual outcome it has obtained. In particular, slotted time $t = 1, 2, \ldots$ is assumed, and that in every time step every agent chooses action a_i^t and obtains payoff π_i^t that depends on its action a_i^t and the actions of other agents $\mathbf{a}_{-i}^t$. After round t agent i compares π_i^t with the payoff it expected to obtain by taking action a_i^t, which is given by

$$\bar{\pi}_i^t(a_i^t) = \sum_{t_{-i}, \mathbf{a}_{-i}^t} \Pr[t_{-i} | \mathbf{a}_{-i}^1, ..., \mathbf{a}_{-i}^{t-1}, \mathbf{s}_{-i}^1, ..., \mathbf{s}_{-i}^{t-1}] \times$$
$$\times p_i(t_{-i}) \times Pr[\mathbf{a}_{-i}^t | t_{-i}] \times \pi_i(a_i^t, \mathbf{a}_{-i}^t)$$

This quantity represents the payoff i expected to make before round t materialized, by summing over its beliefs over its partners' types and expected actions. Hence the desirability of outcome $(a_i^t, \mathbf{a}_{-i}^t)$ is

$$\pi_i^t - \bar{\pi}_i^t(a_i^t)$$

The appraisal variable of control represents whether the agent can achieve high outcomes independently of what other agents do. An agent's "safest" plan is none other than the strategy that will give it the highest value under the worst (adversarial) choice of strategy by the other agents. In the language of game theory, this is often referred to as the "maxmin" strategy, defined as $\text{maxmin}_i^t = \max_{a_i^t} \min_{\mathbf{a}_{-i}^t} \pi_i(a_i^t, \mathbf{a}_{-i}^t)$. An agent is then said to have 'control' in playing a_i^t if

$$\text{maxmin}_i^t \geq \bar{\pi}_i(a_i^t)$$

As an example in a two-player game, consider an agent having two actions, L and R, with L giving it a payoff of 20 no matter what its partner does, and R giving it 40 if its partner also plays R, but zero if its partner chooses L instead. In this game, choosing R constitutes a risky choice, because it might provide the agent with a high payoff of 40, but might also result in a low payoff of zero. If the agent believes with probability greater than 0.5 that its partner will play R then it will take the risk in expectation; otherwise it will stick to the safe maxmin strategy for a guaranteed payoff of 20. In playing R then, and assuming $\Pr[a_{-i} = R] \geq 0.5$, the agent has 'low control' because its alternative payoff (20) is less than the expected payoff under R, and 'high control' if $\Pr[a_{-i} = R] < 0.5$, since in that case the agent might as well switch to its maxmin strategy.

Finally, self-blame models ex-post regret. Having played a_i^t, agent i compares the payoff it obtained with the payoff it would have obtained having chosen an alternative action a'^t_i. If

$$\pi_i(a'^t_i, \mathbf{a}^t_{-i}) > \pi_i(a_i^t, \mathbf{a}^t_{-i})$$

for some a'^t_i the agent experiences regret and blames itself. If $\pi_i(a'^t_i, \mathbf{a}^t_{-i}) \geq \pi_i(a_i^t, \mathbf{a}^t_{-i})$ for all a'^t_i, then the agent does not experience self-blame.

Finally, the above definitions can be modified for agent implementations that do not use utility to encode preferences. For instance, a BDI-type agent might express 'desirability' by comparing its expectations of the state of the world with what has actually happened, 'control' by considering its "safe plan" or worst-possible outcome,

and 'self-blame' by computing appropriate measures of regret.

3.3 Evaluation

To evaluate the affective signaling mechanism, the performance of signaling and non-signaling agents was compared in two domains: a two-agent repeated social dilemma game, and a task domain with collaborative agents, which was modeled as a DEC-POMDP with private utility functions and private state information. In each case, the signaling agents generated and communicated affective signals in every time step, after actions were taken.

In the first domain, agents that exchange affective signals were compared with agents who did not communicate at all. In the second domain, the comparison was extended to include agents that communicated their entire private state information, i.e., all their unobservables. In that domain, the affective signaling mechanism performed as well as full-state communication. But since full-state communication poses an upper bound to the performance benefits that signaling can confer [108], affective signaling should perform as well as (or even outperform) other signaling techniques that communicate less than everything. Therefore, comparisons with other existing signaling techniques have not been part of the empirical investigation.

3.3.1 Simple iterated social dilemma

In the prisoner's dilemma (PD) two agents simultaneously make a decision to cooperate or defect. If both cooperate, both enjoy a high payoff. It is always a dominant strategy, however, for both of them to defect, leaving mutual defection as

the only Nash equilibrium. In the repeated version of this game, the optimal strategy depends on whether the game lasts for a finite number of rounds or an infinite number, with discounting of future payoffs. In the former case, mutual defection in every round remains the only Nash equilibrium, but simple conditions for sustainable cooperation can be derived for the latter case.

In terms of its complexity, the prisoner's dilemma is a game with no uncertainty and is thus of limited interest for signaling. To demonstrate how unobservable characteristics can complicate even in the simplest of games, one small change was made to PD. The payoff structure of some agents in the population was allowed to be different. In particular, some agents were "cooperative" and did not experience a payoff boost from *unilaterally* deviating (see Table 3.3), while some were "self-interested" and had the traditional payoff structure (Table 3.2). Cooperative agents had a payoff structure that is characteristic of the 'Stag Hunt' game, in which mutual cooperation and mutual defection are the two equilibria of the one-shot case.

In this experiment agents maintained individual, and potentially inaccurate, prior beliefs over the distribution of those types in the population. The number of rounds the game would be repeated was finite and sampled uniformly in random from the range $[2, r]$ in the beginning of each game. That number, as well as the value of r, were unknown to the agents. Because agents' payoffs differ, it no longer holds that in a finite-round game mutual defection in every round is the only equilibrium.

Because computing optimal strategies or equilibria in this game is hard in the absence of accurate prior beliefs, agents were allowed to adopt any of a number

of varied strategies. Cooperative agents, who would optimally want to cooperate against cooperators and defect against defectors, could condition their response to whether they believed their partner was also cooperative with probability greater than a threshold θ, the values of which varied across agents ($\theta \in \{0.4, 0.6, 0.8\}$). Self-interested agents could also engage in strategic manipulation, cooperating for a number of rounds λ (the value of which varied from agent to agent ($\lambda \in \{0, 1, 2\}$) before they defected. Each of these strategies performs well against *some* distribution of other strategies, but with the true constitution of the population being unknown none of them can be called truly "optimal."

In the simulations, 1000 agents were paired in random and played $k \sim U(2, r)$ rounds until a total of 20 games were played. The value of r was set to either 3 or 6 for short- and long-running games, respectively. The distribution of agent types, denoting their payoff structure and strategy, was drawn from a hyper-prior to make the initial fraction of cooperative agents more likely to be small (around 25%), large (around 75%) or equal to that of self-interested ones (around 50%). The simulation in every case was repeated 20 times to ensure statistical significance of the results.

	Cooperate	*Defect*
Cooperate	(20, 20)	(0, 30)
Defect	(30, 0)	(15, 15)

Table 3.2: Payoffs for the Prisoner's Dilemma

	Cooperate	*Defect*
Cooperate	(30, 30)	(0, 20)
Defect	(20, 0)	(15, 15)

Table 3.3: Payoffs for the cooperative agent type

In each repetition, the various types of agents were represented in the population in different percentages, but drawn from the same hyper-prior.

Results

The metrics used to assess the usefulness of the signaling mechanism, shown in Table 3.4, capture the accuracy of agents' inferences as well as their performance. The average *belief divergence* of the agents was measured after the first move in each game (β^m), defined for agent i as $1 - p_i(t_j|a_j^1, s_j^1)$, where t_j is the true type of his partner (in the absence of signaling s_j^1 is nil). The average belief divergence at the end of a complete game of k rounds (β^g) was also measured and defined in a similar fashion. For both metrics, smaller belief divergence implies more accurate inference of the type of one's partner. To assess agents' performance, their average payoff (π) was recorded at the end of a game. These metrics were assessed separately for cooperative and self-interested agents. Moreover, the trajectory of the population in an evolutionary framework was assessed using the replicator dynamic [46], by computing the gradient of increase δ of cooperative agents in the population. This measure reveals whether the "good guys" or the "bad guys" will tend to dominate the population in an evolutionary context, and how fast that will come about.

$\beta^m(\mathrm{t}), \beta^g(\mathrm{t})$	Avg. belief divergence over partner's type
$\pi(t)$	Average payoff obtained
δ	Gradient (cooper.) w/replicator dynamic

Table 3.4: Metrics used for the Social Dilemma [subscript (t) denotes measure is broken by agent type]

The results were robust with respect to r (maximum number of rounds in a single social dilemma game) and the choice of hyper-prior (i.e., the average fraction of cooperative agents). For some typical cases ($r = 3$ and p(cooperative) between 0.25 and 0.75), Figures 1 – 4 present the results. Error bars indicate 95% confidence intervals.

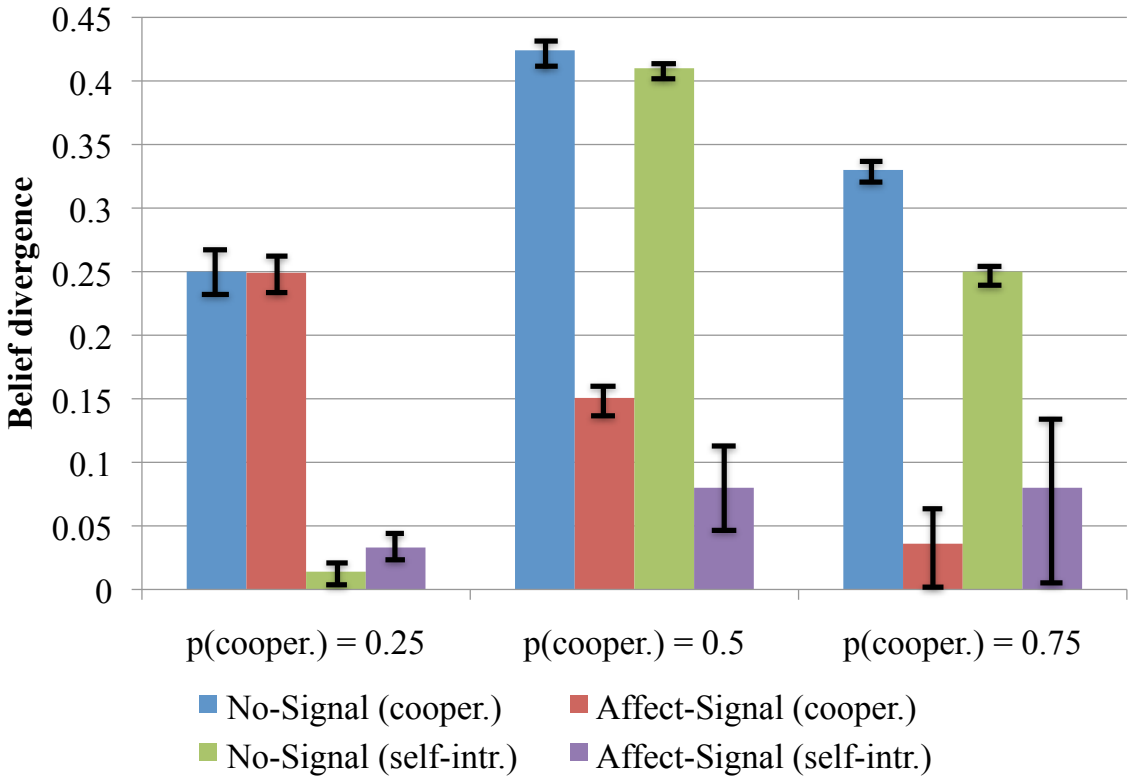

Figure 3.1: Belief divergence (β^m) after the first move (note that $\beta^m = 0$ corresponds to perfect knowledge of the true state of the world)

In most cases, signaling enables cooperative agents to distinguish between a cooperative and a self-interested partner earlier and more effectively (smaller β^m and β^g), which translates to improved payoffs (higher π) and increased selective fitness (higher δ). In fact, cooperative agents, when they are not the majority in the population,

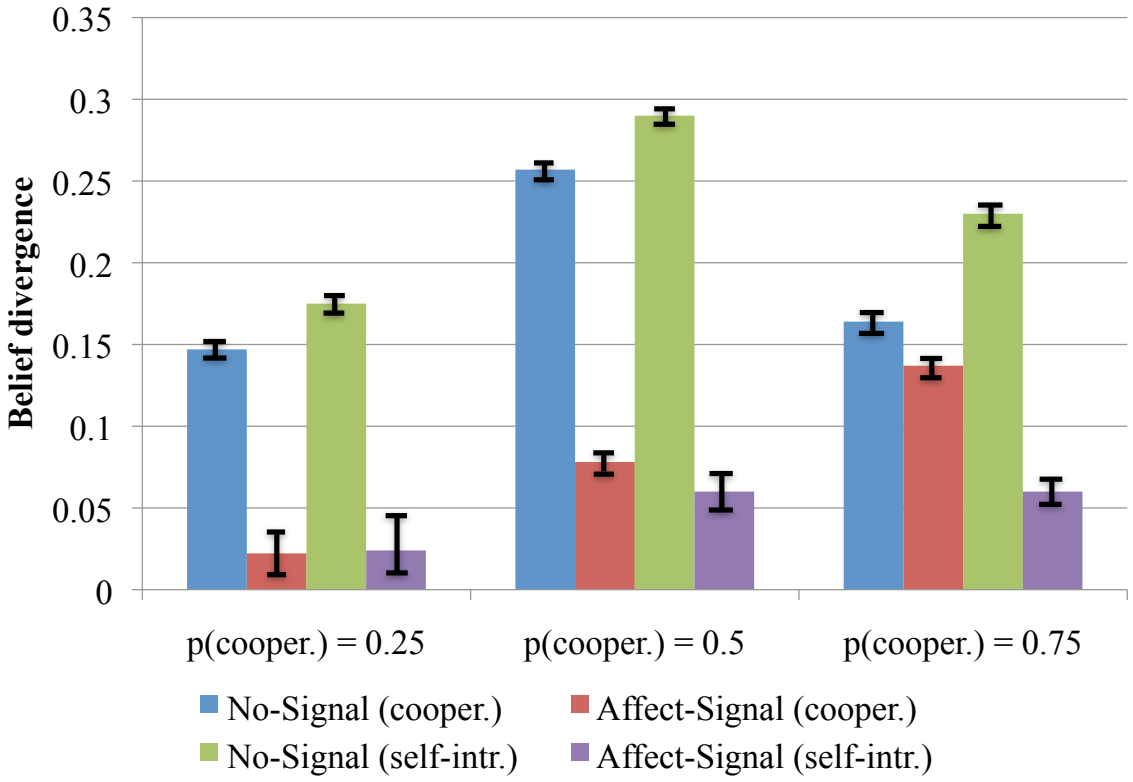

Figure 3.2: Belief divergence (β^g) at the end of a game (note that $\beta^g = 0$ corresponds to perfect knowledge of the true state of the world)

will tend to shrink away without signaling and disappear ($\delta < 0$) but with signaling they dominate the population ($\delta > 0$), as shown in Fig. 4.

3.3.2 Task Domain

The Bank Deposits Task

In the Bank Deposit Task (BDT), two agents, 1 and 2, start from opposite corners in the lower end of a grid world (Fig. 5). Their goal is to collect money, which appears randomly in the upper part of the grid, and deliver it to the bank, at the lowest row of

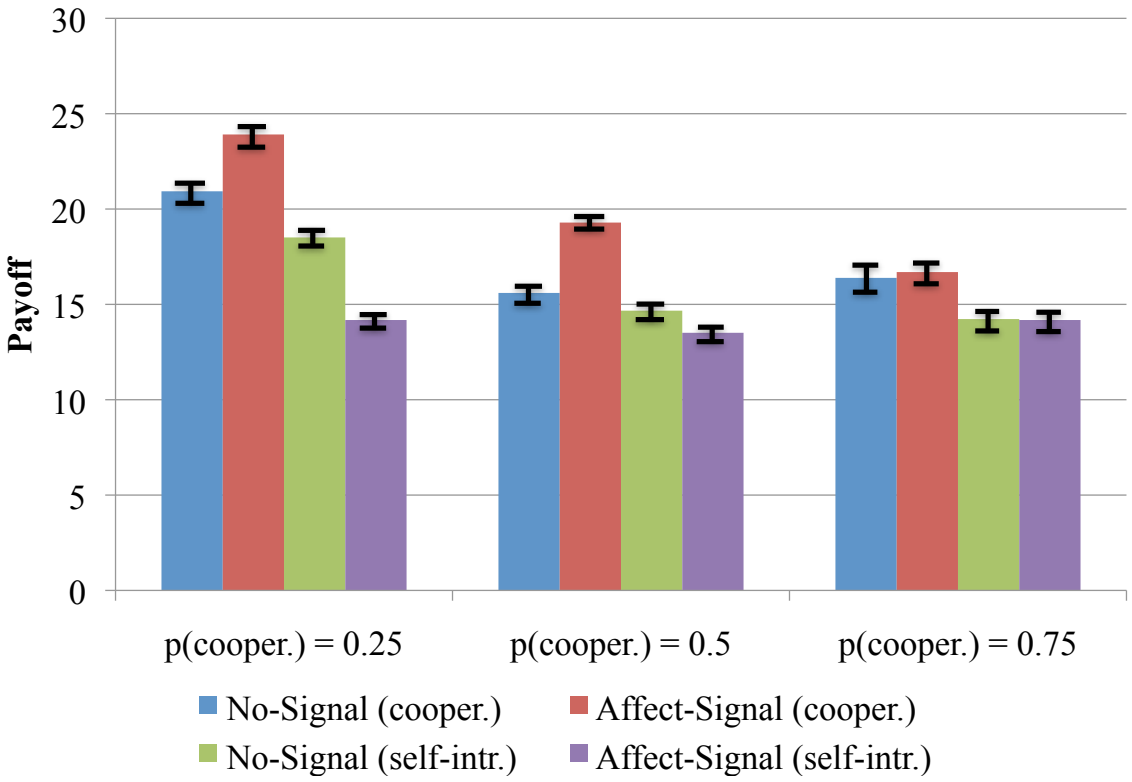

Figure 3.3: Accumulated payoff (π) at the end of a game

the grid. Depositing money in the bank is the only way positive payoffs can be made. To make deposits, agents must avoid bumping into each other (collisions), and avoid enemies that appear and disappear randomly in the narrow corridors connecting the upper and lower parts of the grid. Agents maintain a private state which consists of three binary variables: *broken*, *carrying-money* and *gas-low*. If an agent is broken it can no longer dependably carry out its decisions, i.e., with 30% chance it might move randomly to any position instead of carrying out the action it has decided to do. Both agents may assist each other when they are close (distance ≤ 2) and thus undo the other's brokenness. Agents that are low on gas need to go immediately to

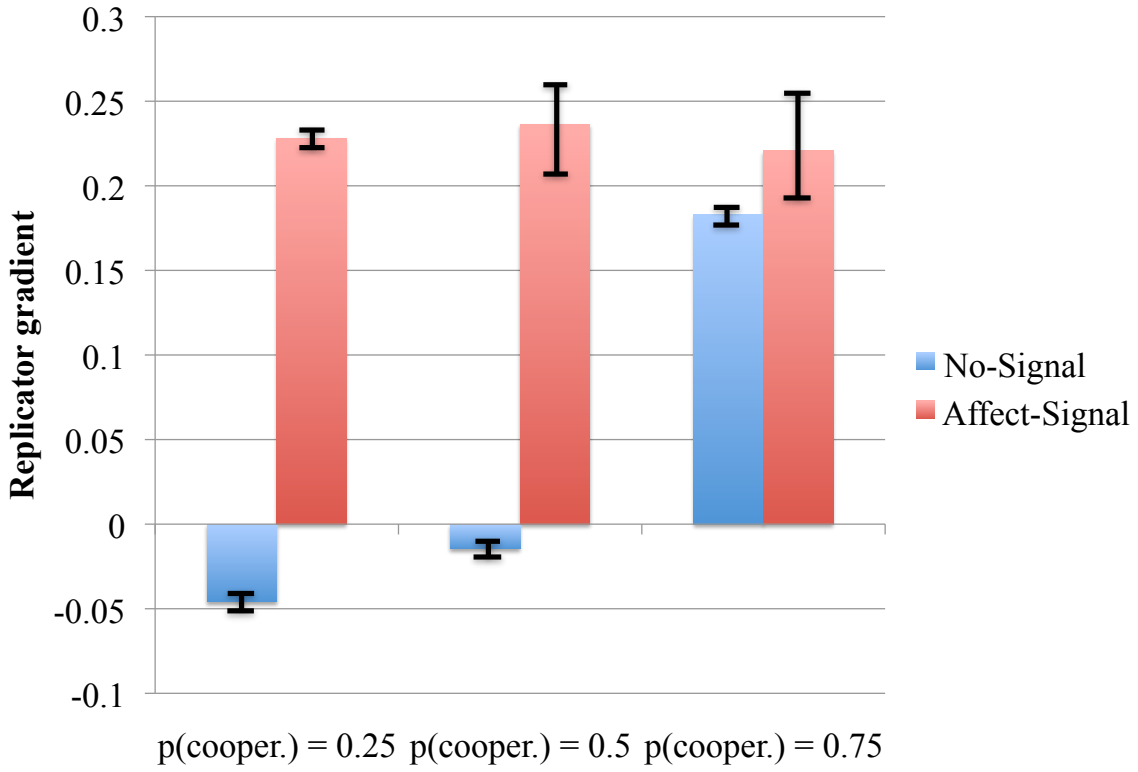

Figure 3.4: Replicator gradient for cooper. agents (δ) (note that a positive δ means that the fraction of cooperative agents will tend to increase over time, while a negative δ denotes that the cooperative type will shrink away)

the gas station (top-right corner) to refill before they continue; they are not allowed to deliver money to the bank while being low on gas. Agents receive a payoff of +100 for a successful deposit, and -5 for every round in which they are broken. Agents are assumed to observe only the space around them (distance ≤ 2).

Modeling BDT with a DEC-POMDP

BDT presents a much richer and more realistic domain than the social dilemma. Agents do not interact in a very "orderly" manner and may take actions that are

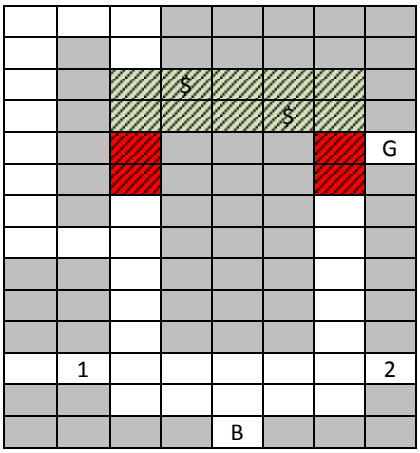

Figure 3.5: DEC-POMDP domain: 1 and 2 represent the initial locations of the two agents, B and G are the bank and gas station, and $ represents the location of money; red and green shaded areas are the possible locations in which enemies and money might appear, respectively; grey-shaded areas represent walls.

not visible by their partner in every step of the game. Stochasticity is introduced with respect to the effects of actions and world state transitions. Moreover, agents now possess unobservable private state information that does not remain fixed but changes across time.

BDT is modeled as a DEC-POMDP. In typical DEC-POMDP formulations, a set of agents N with common utility function u acts in a world which is originally in state s^1. In every time step agents can take actions $\mathbf{a}^t$. After actions are taken, all agents obtain reward $u(s^t, \mathbf{a}^t)$ and the world transitions to state s^{t+1} according to probability distribution $\tau(s^t, \mathbf{a}^t, s^{t+1})$. Agents maintain a common-knowledge prior over s^1, and are assumed to know their common utility function u and the transition function τ. At every time step, each agent i also receives observation $\omega_i(s^t) \sim \Omega_i(s^t)$ which it can use to update its posterior belief over s^{t+1}. The goal of agents in this environment is to maximize a discounted sum of rewards $\sum_{t=1}^{\infty} \gamma^t u(s^t, \mathbf{a}^t)$, in which

$\gamma \in (0, 1)$ is a discount factor. Solving a DEC-POMDP means computing an optimal policy π^* that maps beliefs over states to actions that maximize the above discounted cumulative reward.

The BDT domain differs from this standard formulation in that each agent i has individual utility $u_i(\cdot)$, as its wishes to maximize its own payoff, not the total sum. The fact that agents no longer wish to maximize the same quantity introduces game-theoretic considerations in solving the DEC-POMDP. Second, each agent is allowed to maintain a private state $\sigma_i^t \in \Sigma_i$ that transitions according to the agent's own actions by a function $\tau_i(\sigma_i^t, a_i^t, \sigma_i^{t+1})$ that is common knowledge. This state modulates the agent's own utility function, hence $u_i = u_i(s^t, \mathbf{a}^t, \sigma_i^t)$, but it is private because other agents' observations cannot depend on it. Technically, we require that, if $\sigma^t =< \sigma_1^t, ..., \sigma_{|N|}^t >$, then $\forall \sigma^t, \sigma'^t, \forall s^t, \forall t, \forall i \in N : \Omega_i(s^t|\sigma_i^t, \sigma_{-i}^t) = \Omega_i(s^t|\sigma_i^t, \sigma'^t_{-i})$. Other than that, it shall be assumed that transition functions $\tau, (\tau_i)_{i \in N}$, utility functions $(u_i)_{i \in N}$ and agents' prior over s^1, σ^1 are common-knowledge.

Even for this relatively small domain, the POMDP is prohibitively large. Even with just two agents, the number of possible states exceeds 4 billion. This not only creates problems for pre-computing the optimal policy, but it also makes online belief maintenance and updating onerous. (A proper belief would be a 4-billion-valued vector.) Furthermore, because only bank deposits generate a positive value, payoffs are delayed. Thus, such simple heuristics as 3- or 4-step lookahead do not work well either. Finally, the existence of private state information makes things even harder. To avoid collisions, an agent should know the direction a nearby agent is expected to move. But this depends on whether that agent is broken, carrying money or low on

gas, all of which are part of its unobservable state.

To counter the above limitations an intrinsic reward was added that gave the agent a "fictitious" small reward for moving in the likely direction of money when neither already carrying money nor low on gas, in the direction of the bank when carrying money and not low on gas, and in the direction of the gas station when low on gas. Furthermore, both agents assumed that the other party is using a simplistic three-step-lookahead heuristic to make decisions, and then best-responded to that.

Simulation Conditions

The simulations ran in three conditions: (a) agents observed only each others' actions, without any communication or signaling, (b) agents communicated their entire state to neighboring agents at every time step (a "state dump")[3], and (c) agents communicated affective signals. In all cases, simulations ran 30 times, with each run lasting 40 timesteps.

Results

As before, the measured average belief divergence was measured, defined as $\beta = \mathrm{E}_{t,i}(1 - p_i^t(s^t))$, where s^t is the real state of the world at time t and $p_i^t(\cdot)$ represents i's beliefs at that timestep. Performance-related quantities were also measured, like the average number of rounds an agent was broken, the average payoff at the end of a game, the average number of money units delivered, the average number of times

[3]When an agent received a message containing another agent's beliefs, it tried to *merge* them. To keep matters simple, a liberal approach was adopted: each state in the merged belief set was weighed by its probability in the two original sets.

attacked by an enemy, the average number of times an agent chose to assist the other, and the average time of collisions between agents.

As shown in Table 3.5, once again affective signaling improves performance compared to not signaling. The most interesting result of this experiment, however, is that affective signals improve payoffs essentially as well as communication of full state information, despite being much less informative than a "state dump" and even though agents' belief divergence is as great for agents using affective signaling as for agents that do not communicate. The main source of this promising result is that affective signals efficiently indicate a key unobservable, namely whether an agent is broken or not. Earlier detection of brokenness allows agents to assist each other sooner and thus improve their payoffs. Brokenness is a low-probability event and merely observing a nearby agent's actions does not provide sufficient evidence to boost its posterior probability. Affective signaling thus decreases belief divergence for a crucial variable, communicating effectively information that is highly relevant to successful task performance.

Affective signals are not helping agents infer just whether the other agent is broken. They also communicate other information valuable to their decision-making. As shown in Table 3.5, affective signals are successful in signaling the presence of a nearby enemy. Inferring the location of enemies before they are encountered significantly improves an agent's payoff, as it reduces the number of times it gets attacked. On the other hand, other aspects of the state, such as whether the agent is carrying money or is low on gas, do not elicit unambiguous affective signals, hence they are not as quickly inferred by the other agent when affective signals are being used, compared

to full-state communication. Quickly and accurately inferring these two unobservable characteristics would make the other agent's movement more predictable. (For example, an agent who is low on gas can be safely predicted to move towards the gas station in the next round.) As shown in Table 3.5, agents exchanging affective signals are three times more likely to engage in a collision with each other, compared to agents communicating their entire private state information.

	No-comm	Full-state	Affect
β	0.629 ± 0.03	0.37 ± 0.08	0.603 ± 0.03
Payoff	-180 ± 34.2	-79 ± 29.1	-55.5 ± 94.2
Broken	38 ± 0.0	8.8 ± 4.96	12.1 ± 6.57
Dropoffs	1.9 ± 0.22	0.25 ± 0.23	2 ± 0.2
Collisions	0.55 ± 0.22	0.3 ± 0.13	0.9 ± 0.13
Attacks	0.45 ± 0.13	0.9 ± 0.64	0.05 ± 0.09
Assists	0.0 ± 0.0	1.7 ± 0.26	1.9 ± 0.19

Table 3.5: Results for POMDP simulation (ranges reflect 95% confidence intervals)

3.4 Signal Truthfulness and Manipulability

One concern about signaling is whether it is truthful. The results have presumed that agents are always truthful, and would not hold otherwise. This is because the signal generation is unintentional and automatic, and hence not the subject of deliberation and strategizing. However, even if agents *could* deliberate, computational arguments might support the assumption of signal truthfulness in many complex domains.

To get an intuitive sense, imagine the space of strategies S an agent may adopt for a particular game. For most realistic, non-trivial games, identifying the optimal strat-

egy in S is thought to be a problem requiring exponential time, and as a result agent designers usually have to settle for approximations or—more typically—heuristics that are shown to perform well in practice. In such settings, adding signaling would increase the strategy space to $S \times M$, where M is the set of signaling strategies (what to signal in every contingency in the game). If then computing the optimal strategy is difficult in S, it will be just as hard—if not harder—in $S \times M$. Hence, although agents *could* lie about their sentiments, it might be too hard to know *how* to lie. Similar arguments about the hardness of lying have been invoked in the field of mechanism design [104, 127]. On the other hand, for agents to be truthful, they just have to compute simple quantities as mentioned in Section 3.2.1, which is computationally trivial.

The above argument rests crucially on the assumption that one needs a successful fabrication of the communicated signal to do well, and this fabrication is hard to find. If communicating something in random is successful enough most of the time, then identifying the optimal fabrication, no matter how hard, might be unnecessary in practice. That said, the argument carries extra strength in the face of uncertainty. Lying about your attitudes in a certain way might be effective against some other types of agents, but will likely *not* be effective against populations of *any* composition. Moreover, a "consistency check" might help identify and marginalize liars. As suggested by Scharlemann et al. [109], lying about one's emotions is not hard in one-shot, short-term interactions. The difficulty emerges in long-running interactions, in which the liar needs to be consistent with his lies and with his actions. If then truthful agents in the population adopt a punishment strategy that avoids interac-

tions with any agents that have been "inconsistent" even once in the past, the liars' ability to thrive will be curtailed. I acknowledge that the above argument is largely a conjecture that will be the object of future study to more formally (or empirically) demonstrate its truth. I do argue, however, that it suffices for my truthful-signaling assumption to be defensible in practice, even if agents could deliberate about what to communicate.

3.5 Discussion and Extensions

The work presented in this chapter touches on many strands of work in computer science and psychology. To generate the signal, cognitive appraisal theories [73, 41] have been leveraged. These theories treat emotions as a cognitive and physiological phenomenon that rests predominantly on information processing. In particular, emotions arise when the person evaluates a stimulus (event in the world, or thought) with respect to her goals. The different evaluations (appraisals) correspond to different emotions. For instance, 'fear' is generated when a stimulus is evaluated to be treating to an an important goal, and the person is not situated well to address that threat (low power). If the person is situates well to address the threat, 'anger' might emerge instead toward the source of the threat.

Such cognitive appraisal theories led to the development of computational models of emotion [50, 118, 133]. These models adopt the information-processing view of the underlying psychological model, but aim to precisely define and quantify the notions that psychologists often avoid specifying exactly [81]. The affective signaling mechanism presented here is not a fully-developed computational model of emotion, as it

does not prescribe what emotion the agent will experience under every circumstance. Moreover, computational models of emotion are evaluated as to how well they replicate, explain or represent typical human emotional responses to stimuli. The signaling mechanism is on the other hand exclusively designed for use by artificial agents and makes no effort to emulate people's rich emotional repertoire.

The role of emotions and their expression in decision-making has been studied in game theory [18, 98], in psychology [27] in AI [3], human-computer interaction and in the design of virtual agents, as well as studies of negotiation [1] and social dilemmas [30]. In particular, the prisoner's dilemma has been used in countless studies as a paradigm for cooperation or the tradeoff between individual and group benefits [72], as well as in evolutionary settings to establish the emergence of cooperators [8].

Signaling has been investigated as a facilitator of cooperation in psychology [40, 15]. In computer science, signaling has been studied in the subfields of planning [135] and especially in POMDP-type settings [122, 134, 108, 126]. Using emotions to improve agent coordination has been suggested by Gage et al. [43]. In it, robots acquire resources and express their attitudes in order to avoid conflicts, which is similar in motivation to this thesis' contribution. The emotion generation in their work is not based systematic appraisal mechanisms, and thus not general-purpose, i.e., it is not clear how the same mechanism could be used in a different domain. Affective signals have also been used for computer-human interaction [112], and humans have been shown to distrust agents that do not signal [136].

A very important difference between the model presented in this thesis and other signaling models is that signals are generated non-intentionally. In other words, the

agents do not just "speak by doing," nor do they just "do by speaking," since they never reason strategically about their communicative acts. Rather, their signals are being produced by means of a reflective, appraisal-based, automatic and unintentional subsidiary system. Despite the signals being unintentional, however, they are useful in getting good performance out of the agents using them.

The notions presented in this chapter could be further developed. First, one may choose to enrich the signaling model in order to capture emotional expressions common in humans but absent in the mechanism currently, such as boredom, arousal, surprise, and fear. It is an open question which types of emotional communicative acts will be useful in practice, and which domains might they might offer an advantage in. Second one may investigate whether agents that compute and communicate these particular affective signals can be more successful in their interaction with humans. The argument here is that emotion-like communicative acts carry a lot of meaning for humans, and are intimately associated with perceptual and cognitive interpretations people frequently use. For instance, an angry face of a random person on the street instantly evokes tacit and and explicit knowledge regarding that person's background, desires and intentions. Hence, using communicative signals in human-computer interaction might have the potential to alter human perceptions and influence their behavior. This is the topic of the next chapter.

Chapter 4

Interacting with Computers: Emotion Expressions

The two previous chapters have demonstrated the benefit that comes from incorporating concepts typically associated with human emotions into the design of computer agents. Agents using affect-like computational operators perform have been shown to perform well in complex domains. Automatic unintentional affective signals, on the other hand, can facilitate quicker and more accurate inferences and improved coordination within population of collaborating agents.

So far the systems that have been examined consisted exclusively of computer agents. When humans are also active in the world, however, several important aspects of the system are significantly altered: First, people make decisions in ways that are not completely under the control of the system designer. Human decisions have also been shown in the literature to not strictly adhere to any normative theory (utility theory, game theory) and to be influenced by a variety of cognitive biases and

heuristics, as well as psychological cues, social norms, their mood, as well as their physiological state (hungry, cold, etc.).

These aspects of human decision-making must be considered for designing successful computer agents. The operational efficiency and ultimate success of a mixed system (consisting of both human and computer actors) depends crucially on the way people in it will behave. As a result, it becomes critical that computers engage and interact with humans in ways that promote good and efficient behaviors. To achieve this, computers often need to reason about the ways people make decisions and perhaps communicate with them using cues that people intuitively understand and respond to.

This chapter presents one study demonstrating that *emotion expressions*, when used appropriately by computer agents, may influence how people perceive these agents. In particular, a computer agent in this study is negotiating with a person on how to split a set of resources. During the negotiation the agent is expressing emotions on a virtual face, namely anger or joy. After the game is over perceptions of the agent's trustworthiness by its human partner are measured by means of a separate "trust game." The study's main finding is a *consistency effect*. People seem to trust (and prefer to interact again with) agents whose emotion expressions reflect the same attitudes as their actions in the negotiation game. For example, among "tough" negotiators those who express anger are preferred for future interactions over those who smile, whereas among flexible and cooperative negotiators those who smile are preferred. A possible explanation of this finding is that emotions expressions that are consistent with actions make agents more predictable and easier to understand, hence

"safer" as future partners. On the other hand, agents whose behavior does not "make sense," in the sense that their expressions and their are reflecting contradictory goals, are being perceived as "wildcards" and are thus avoided.

This study builds on the principle that computers are perceived as social actors. Section 4.1 thus presents empirical findings that support the idea that people treat computers as if they had goals, intentions, emotions and social skills similar to humans'. Section 4.2 describes the study and presents its results. Finally, Section 4.3 discusses why emotion expressions are a powerful tool in computer-human interaction and outlines possible future uses for them in a variety of domains.

4.1 Computers as Social Actors

Clifford Nass et al. [94] were the first to discover how people interact with computers in fundamentally social ways. Their main finding was that people's social responses to computers are not the result of a conscious belief that computers are human (or human-like). Furthermore, people's behavior is not a consequence of their ignorance or any psychological or social dysfunction, nor do they think they are indirectly interacting with the programmer of the system.

There are numerous studies in which people have been demonstrated to treat computers as social actors. When users are asked to grade the performance of a computer program, they are significantly more lenient when they submit their report to the same physical machine on which the program ran, and more objective when they submit their report to a different machine. In this way they mimic the broadly acceptable social behavior of being more objective when writing a recommendation

letter for someone, as opposed to offering them a face-to-face evaluation. Similarly, when computers were programmed to praise another software program, if the praising computer was running on a different machine than the program being praised, people considered this praise as more significant. In this way they mimic the social rule that being praised by others is more genuine and more significant than someone praising themselves.

Gender stereotyping is another way in which people anthropomorphize computers. Social psychology indicates that people generally find praise from males more convincing than praise from females, and that males who praise are more likable than females who do so [94]. In a study in which computers were given a male or female voice to suggest a notion of gender [74], people reported liking the computer with the male voice more when it praised them, and they also founds its praise to be more significant. Moreover, computers that talked about technology were found to be more knowledgeable when they spoke in a male voice, whereas those that talked about love and relationships were perceived more knowledgeable when they spoke in a female voice, another instance of gender stereotyping.

A number of other studies have demonstrated similar effects. When a system was installed in cars that monitored the drivers' behavior and warned them of dangerous actions, they became annoyed and drove even more carelessly [52], often citing that the computer in their car had become a "backseat driver."

Many of the studies have revealed that the quality of the interaction correlates with the degree in which the computer's behavior *matches* that of humans. In interactions between humans, for instance, people respond more favorably to other who

are like them. In one experiment [95] people were shown items on sale on the eBay auction site and were asked how much they would pay for the goods. Both sellers and buyers were given personality tests to determine whether they are introverts or extroverts. It was then shown that introverts would agree to pay higher prices for items whose description had also been written by an introvert, and likewise for extroverts. In computer-human interaction a similar effect was shown. In the "Desert Survival Situation" (DSS) task[1], which has gained popularity among social scientists, people are given feedback about their choices from a computer in a way that reflects a "submissive" or "dominant" personality. People found the computer whose personality traits matched theirs to be more intelligent and more insightful in its feedback.

In another study [93] that illustrates the importance of *matching*, drivers were primed to experience joy or sadness before asked to complete a driving run. The cars they used were equipped with a voice-generation system that gave them instructions, and its voice was manipulated to sound happy or sad. Subjects whose emotion matched the one their car's voice exhibited not only reported higher satisfaction levels but also performed better as drivers (fewer accidents).

Matching in this case not only refers to an alignment in the mood or personality between the human and the agent. It might also refer to congruency between he agent and the content. In one experiment [92] subjects had an agent tell them a story. The story was either happy (a cure for cancer) or sad (a spate of dead gray whales), and the agent reporting it sounded either joyful or morose. People rated the happy story as more joyful when it was delivered in a joyful manner, and the sad story as more

[1]In this task participants are told they are stranded in the desert and they must pick a small number out of several items (a knife, batteries, a lighter, etc.) to ensure their survival.

saddening when delivered in sad voice. On the other hand, when the mood of the story and the tone of voice of the agent were misaligned, subjects found the emotional content and the relative significance of the story to be both less pronounced. In other words, an affective match or mismatch between mood and content influences how that content is perceived.

The surprising thing about all these studies is that people do not perceive the human characteristics (such as gender) of the person "behind the computer." As a matter of fact, in another study [115], a confederate was present during the computer-human interaction and referred to the computer as 'the computer' or 'the programmer,' while the computer referred to itself as either 'the computer' or 'I.' There was no significant difference in the way in which people behaved in each condition, which implies that social attributions are made directly to the computer, not the person (designer, programmer) that is indirectly responsible for the computer's behavior.

People's behavior toward computers exhibits even stronger social characteristics if the computer is represented as a *virtual agent*. Virtual agents are typically 3D-rendered human-like characters with a face and sometimes a body. They are usually capable of directing their gaze, expressing emotion through their face and posture, synthesizing speech, and in some cases even having conversations with people, by parsing human speech and utilizing a knowledge base containing facts, beliefs and social norms. Several studies have demonstrated the power behind virtual agents. People are more likely to reveal private information about themselves when paired with a virtual agent that builds trust between itself and its human partner [51]. People attribute intentions and personality traits to virtual agents in the context

of a prisoner's dilemma game [32] or a negotiation [31]. Cultural concepts such as religion or moral values are invoked by the apparent ethnicity or social background of the virtual agent [33]. Virtual agents have also been used to gauge human behavior in critical but otherwise unlikely circumstances (such as an emergency evacuation), which was later used to design better evacuation policies that are aware of people's responses under extreme stress [68]. Many interesting psychological interventions have been made possible by the use of virtual agents as well. In one experiment it was shown that veterans suffering from Post-Traumatic Stress Disorder (PTSD) experience reduced negative symptoms when given the chance to interact frequently with virtual agents that "understand them" [67]. Such agents have also played the role of the patient, helping young therapists gain experience before they apply their skills and practice to real people [66].

In the study described in this chapter people interacted with a virtual agent who could display emotions on his face.[2] People negotiated with the agent over how to split a set of items, and the agent would in some cases express joy or anger at the person's actions. After several such games were played with different agents that used the same negotiation strategy but expressed different emotions, subjects had to select one of these agents to play a 'trust game.' People's selection of a partner for the trust game was used as a metric of that agent's perceived trustworthiness. The main finding of the study was consistent with prior *matching* insights discussed above. In particular, among agents that employed a "tough" negotiation strategy, people preferred angry agents over smiling ones. On the contrary, among agents who

[2]To simplify the design, all the virtual agents were male, caucasian (hispanic or non-hispanic) and in their twenties.

were flexible and concessionary negotiators, subjects preferred smiling over angry partners. When thus the agent's emotions expressions *matched* its actions, the agent was perceived to be more trustworthy.

4.2 Perceptions of Trustworthiness

The term *trustworthiness* usually carries a double meaning: it may refer to someone's *ability* (or skill) to carry out a task, or their *intentions* (or motive) for doing so. For example, a patient might trust her doctor because (a) he possesses the necessary training and experience to perform a sensitive operation, a matter of ability and skill, and (b) he desires for the patient to get better, a matter of intention. Similarly, someone might trust her husband because he possesses both the self-control (skill) and the desire (motive) to remain faithful to her.

People's perceptions of agents' trustworthiness are significant, as they influence their behavior towards the agents, and may foster or inhibit repeated interactions between them. Trust reduces the transaction costs of economic and social exchanges among humans [19]. Interactions with our friends are easier, as we trust their words and feel less inclined to thoroughly verify their statements, because we trust them. Business partners that have developed mutual trust can perform economic transactions faster, without excessive documentation to safeguard against fraud. The economic benefits of trust have been documented in the Internet economy as well. In one experiment [9] it was shown that eBay sellers with a high positive rating—a proxy of trustworthiness—could on average command 8% higher prices over sellers who lacked high ratings. As a result, whether agents can cause people to trust them

is an important aspect of their design.

To study how perceptions of agents' trustworthiness are formed in people, a negotiation game was used. Negotiation is a commonly-used method for parties with diverging interests to reach a mutually-beneficial agreement. People negotiate over how to schedule activities when participants have different time constraints and priorities, to efficiently allocate valuable resources across individuals or corporations with varying needs and preferences, or to resolve international conflicts without resorting to violence. The significance of negotiation has led to a large literature in the fields of psychology, economics, sociology and computer science [76, 106, 60]. Computer agents have been used in many negotiation settings, sometimes acting on behalf of humans [60] and sometimes negotiating with them [77]. As the scale and complexity of the domains in which negotiation is employed are expected to increase, the use of computer agents as negotiators might grow. Examples of such domains might include traffic management [63] and commerce [80], among others. Furthermore, computer agents have shown potential (compared with human negotiations) for improving negotiation outcomes in some cases [78].

Yet negotiation studies with computer agents have largely overlooked the fact that humans use significant verbal and non-verbal cues when they negotiate [35]. The expression of emotion, in particular, has been shown to significantly influence negotiation outcomes [10, 131]. For instance, displaying anger was shown to be effective in forcing larger concessions out of the other party, whereas positive emotion was found to be helpful in exploring and achieving mutually beneficial (integrative) solutions [130, 20]. The effects of emotion during business negotiations were also shown

to be modulated by culture [75]. In most studies in which emotion was expressed by computer agents, this emotion was conveyed by means of text sent by the computer agent to its human partner [130] the computer would say "this offer makes me really angry"). More recent implementations of virtual agents tested for differences among different modalities of emotion expression (e.g., text or virtual face) and found no significant differences [32].

Equally important is the fact that negotiation can distinguish between the two meanings of trustworthiness. As discussed below, subjects in the study negotiated with various computer agents on how to split a set of resources (virtual coins). After the negotiation was over, they had to select one of these agents to play a *different* (trust) game, in which the agent's motives and intentions mattered, but not its skills and abilities. In other words, people received information about both the agents' negotiation skill and their motives during the negotiation games, but only the latter were used in the evaluation of trust.

4.2.1 Experiment Design

To investigate the effect of emotion in human-computer negotiation and perceptions of trustworthiness the following game was developed: A human subject (h) is paired with a computer agent (a), and they must negotiate on how to divide a set of resources amongst themselves. The resources consist of virtual "coins" of four types: gold, silver, bronze and iron. In each game, there are three (3) coins of each type. Before the game starts, people are told the *relative* value of each coin type, i.e., that gold coins are more valuable than silver, which are more valuable than bronze, which

are more valuable than iron coins. However, people are not given the exact numeric value (in points) of each coin type.[3] Subjects are also informed that the relative valuation of the items by the agents might be different than their own, e.g., computers might prefer silver coins over gold ones. Notationally, the four item types are referred to with numbers $j \in \{1, 2, 3, 4\}$. The notation w_j^i is also used to denote the number of points player $i \in \{h, a\}$ receives by possessing a coin of type j. In all experiments $\mathbf{w}^h =< 10, 7, 5, 2 >$ and $\mathbf{w}^a =< 10, 2, 5, 7 >$. Notice how item types 1 and 3 (gold and bronze) present a "distributive problem," i.e., players need to decide how to split items of common value, but items 2 and 4 (silver and iron) present "integrative potential," i.e., there are exchanges of items that lead to mutual benefit. Moreover, it must be pointed out that computer agents have full knowledge of vectors $\mathbf{w}^h$ and $\mathbf{w}^a$.

The game proceeds by means of alternating offers, and participants play in turns, with the human always making the first offer in a game. An offer by player $i \in \{h, a\}$ consists of a complete allocation of all coins between the two participants. The notation $c_j^i(t)$ is used to denote how many items of type $j \in \{1, 2, 3, 4\}$ player i was given in the offer made at round $t \in \{1, 2, \ldots\}$. Hence, allowing only for complete allocations means that in every round t, offers must satisfy $c_j^h(t) + c_j^a(t) = 3, \forall j$. In every round $t > 1$ the negotiator whose turn it is may *accept* an offer, in which case the game ends and both participants make their corresponding points; for the human player, these would be $\pi^h = \sum_j w_j^h c_j^h(t-1)$. Alternatively, she may *reject* the offer, and counter-offer a different split of the items. Finally, at any point in the game,

[3] A preliminary experiment with $N = 25$ showed that people's behavior is not affected by them knowing the point value of every coin type.

either participant may *drop out*. A game consists of a maximum of 15 rounds.[4] If no agreement is reached in any of these rounds, or if either player drops out, both players make zero points.

The agents in the experiment differed in two ways: with respect to their *strategy*, and with respect to their *emotion expression*. The strategy of an agent encompasses when offers are accepted or rejected, and what counter-offers are made by it. Likewise, the emotion expression of an agent defines whether emotion is expressed, and what type of emotion is displayed in each circumstance. Below the strategies and emotion expression policies used in the agents are presented; the way in which emotion was displayed to the human is also discussed.

Strategies of computer agents

The strategy of an agent prescribes how the agent behaves as a negotiator. Although the literature on effective negotiation strategies is extensive [125], straightforward, intuitive strategies for this experiment were used to keep matters simple. The goal was not to exhaustively explore the effect of emotion expression given complex strategies, but to assess whether emotion has any effect on people's behavior in computer-human negotiation, and whether this effect is dependent upon the agent's strategy.

The strategies varied along two dimensions: "flexibility" and "self-interestedness." An agent follows a *flexible* strategy if its offers change from round to round throughout

[4]Notice how, if the human always makes the first offer, the agent always makes the last offer. If the game reaches the 15th round, then the human can either *accept* the computer's offer, or *drop out*, since there are no more rounds for counter-offers to be made.

the game; its strategy is *inflexible* if it always makes the same offer (or very similar ones) in every round. In a similar fashion, an agent is said to follow a *self-interested* strategy if it attempts to keep for itself almost all the points being negotiated; its strategy is *non-self-interested* if the agent seeks agreement on balanced offers, which provide a more or less equal split of the points. Four simple strategies were used, which are described below. (Table 4.1 groups them according to flexibility and self-interestedness.)

1. *Selfish:* The selfish agent in every round chooses a single coin at random (but never a gold one) to counter-offer to the human, and keeps everything else for itself. Thus selfish agents are inflexible and self-interested.

2. *Nash:* This agent computes the Nash bargaining point (N.B.P.) of the game, which is the allocation that maximizes the product of both players' payoffs, and offers that in every round. The N.B.P. presents the theoretically most efficient point in the negotiation, as it is Pareto-optimal and satisfies a series of axiomatic constraints [90]. N.B.P. allocations split the points in a very balanced fashion, thus this agent is inflexible but non-self-interested.

3. *Conceder:* This agent performs concessions in a constant rate. In particular, no matter how the human behaves, at round t the agent offers her $\frac{3t}{2}$ points and keeps everything else for itself. In other words, the first time it plays it will offer the human 3 points (round 2), the second time 6 points, etc. Since this agent starts from very imbalanced offers (only 3 or 6 of a total of 72 points) and concedes slowly, it is categorized as self-interested but flexible.

4. *Tit-For-Tat:* This is an agent implementing reciprocity. In round t it offers the human $0.8 \times \sum_j w_j^a(t-1)$ points. Hence, the more concessionary the human has been in her last offer, the more concessionary the agent becomes; likewise, if the human has been selfish, the agent would reciprocate this. The 0.8 coefficient represents a degree of "toughness" by the agent, i.e., it reciprocates slightly less than what it is being offered. This agent is both flexible and non-self-interested, as agreement will only be reached when the two players start conceding, eventually "meeting" somewhere in the middle.

	Inflexible	**Flexible**
Non-self-interested	Nash	Tit-For-Tat
Self-interested	Selfish	Conceder

Table 4.1: Strategies used in the negotiation game, grouped according to their flexibility and self-interestedness.

All agent types accept an offer made by the human if and only if the points they would request in their counter-offer (according to their strategy) are no greater than the points the human is currently giving them. Agents never drop out of the game. Also, whenever agents wish to give a certain number of points to the human, they choose the most integrative way of doing so (i.e., of all possible counter-offers that would give the human c points, they choose the offer that maximizes their own points).

Emotion expression by agents

The emotion expression policy of an agent denotes whether and how it displays affect. Affect in the game was displayed on a "face" the agent was given. Faces

were all male, and were randomly assigned to the various agents from a pool of 15 templates, such that no subject would interact with two agents bearing the same face during the experiment. The face of the agent was rendered to the side of the game board, on which the items were being negotiated. Five emotion expression policies were used, described below:

1. *No-Face:* This is the baseline case, in which there is no visible face to the agent.

2. *Poker-Face:* This agent shows a face, but never displays any emotion on it, always keeping a neutral expression. This agent was used in addition to the No-Face agent to assess how much of any effect comes from displaying emotions, or merely from the presence of a face (even if it displays no emotions).

3. *Always-Smile:* This agent displays a face and smiles to all the offers made by the human, independently of what these offers look like.

4. *Always-Angry:* This agent displays a face and expresses anger toward all the offers made by the human, again, independently of what these offers look like.

5. *Appraisal:* This agent would smile or show anger depending on the human's actions, instead of following a fixed strategy. If at round t it was offered by the human at least $\frac{3}{2}t$ points it would smile, otherwise it would show anger.

All agents that display emotion follow the same pattern of expression: First, they "look" to their right, where the coins are, to "see" the offer made by the human; they then "look back" toward the human (straight gaze), perform their expression (of joy or anger), and send their counter-offer (or acceptance notification). Joy is expressed by

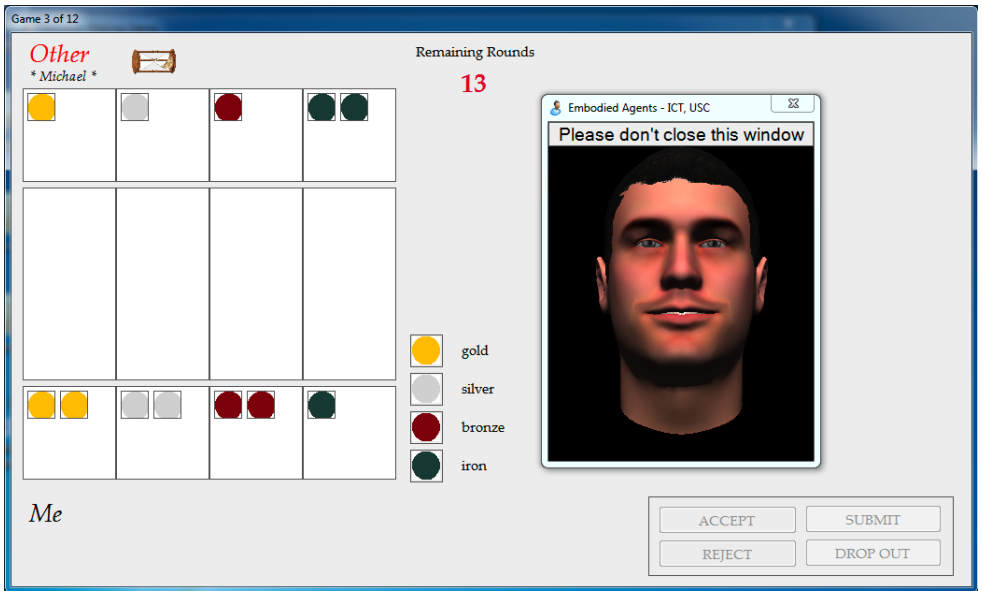

Figure 4.1: The Negotiation Game

a smile across the face, which forms in moderate speed (1 sec). Anger is expressed by contraction of the corrugator muscle (frowning) as well as an aggressive twitching of the mouth. Expressions dissipate linearly towards normal (expressionless face) after the counter-offer is sent, while the human is deciding her move. These particular displays were shown to be successful in conveying the desired emotions in [32]. Also, no "gradients" of the expressions were employed (e.g., more or less intense smiles)—all expressions were of the same intensity. A screenshot of the negotiation game can be seen in Figure 4.1.

It must be pointed out that several factors in the presentation of emotion have been overlooked in order to keep the experiment simple, although they could presumably be carrying strong effects which are well-documented in the literature. In particular, gender effects were not tested for, as all the agents were male. The effect of race,

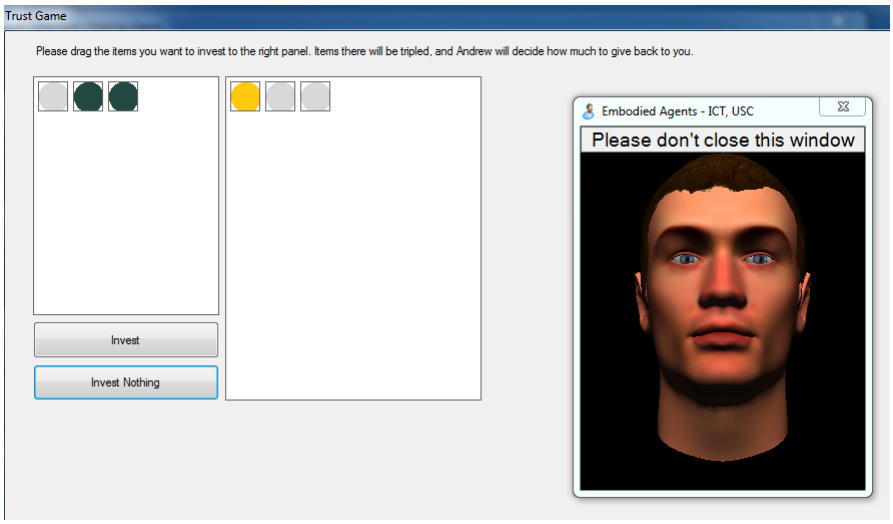

Figure 4.2: The Trust Game

age, or physical attractiveness was also not examined, although all of these could mediate how expressed emotions are interpreted. In all these cases, all these variables were kept constant (using agents of the same age and gender) whenever possible, or randomized uniformly (for race).

The trust game

Each subject in the experiment plays several negotiation games with different agents. After every three such games, in which the agents differ in their emotion expression policy, but not in their strategy, subjects are asked to play a 'trust game.' As an example, someone might play with three Tit-For-Tat agents, the first of whom always smiles, the second always shows anger, and the third maintains an expressionless face. After these three games, the subject is asked to play a trust game with one of them. The trust game is a variant of the popular public goods game. To

play it, people first have to select which of these three agents they want to be paired with. (Agents are given names at random, like "Michael" or "James" to assist recall.) After they indicate their choice, they are presented with a *trust* problem. In particular, they are shown their total gains (in coins) from the previous three negotiation games, and are asked what fraction of these they are willing to trust to the agent they have chosen as their partner. If the agent's policy includes showing a face, it also is displayed, but no emotions are ever shown on it.

If the subject chooses to trust her partner with a non-empty subset of her gains $\mathbf{t} =< t_1, t_2, t_3, t_4 >$, where t_j is the count of coins of type j she has trusted, this subset is tripled at the hands of the agent. Then, however, the agent chooses what percentage $p \in [0, 1]$ of the value of these coins it will return to the human, and how much it will keep for itself. Agents all choose p uniformly at random from $[\frac{1}{3}, \frac{2}{3}]$, but subjects are told nothing about this. The subject at the end of the trust game keeps the points she has not trusted to the agent, to which $p \times \sum_j (3t_j)w_j^h$ points is added. The trust game looks like the screenshot in Figure 4.2.

Experiment process

Each subject in the experiment played twelve negotiation games, in sets of three. Each triad involved computer agents that differed in their emotion expressions but used the same negotiation strategy, and was followed by a trust game. The order of games was randomized, and each subject faced triads of agents that differed in terms of strategy of emotion expression policy, but not both. Instructions were delivered to the subjects over video, and they were all debriefed after the end of the experiment.

After each negotiation game, subjects were asked to answer, in Likert 1-7 scales, four short questions regarding their experience with the agent they had just played with. Subjects were paid $20 for their participation, which lasted about 45 minutes. They were also told that the person who would score the highest number of points would be awarded an extra $100. The total number of subjects was $N = 88$, for a total of 1,056 negotiation games and 352 trust games.

4.2.2 Hypotheses

Two hypotheses were examined in this study. The main hypothesis concerns the influence of emotion on perceptions of trustworthiness. Also, since there are plenty of reported results in the literature, a second hypothesis was added, namely, to replicate and confirm some of the best-known findings regarding the influence of agent emotion expressions on people's negotiation behavior.

The main hypothesis (H1) is formulated using the notion of *action-consistency*. An emotion expression is called *action-consistent* with a strategy if the emotion emphasizes the characteristics of the agent that manifest in its actions. Positive emotion typically emphasizes kindness and goodwill, whereas negative emotion is usually illustrative of selfishness and intransigence. Hence, positive emotion is more *action-consistent* with non-self-interested strategies, and negative emotion is more *action-consistent* with self-interested strategies. In the same spirit, positive emotion is more *action-consistent* with flexible than with inflexible strategies. Alternative notions of consistency, which are not discussed here, have been introduced in the literature before [91].

Note here that the definition of action consistency employs the "typical," most common, and context-independent meaning of emotion expressions. Hence, a smile might be typically reflective of good intentions and kindness, but in certain contexts it might be a sneer, an indicator of Schadenfreude (rejoicing at the misfortune of others), or simply a polite gesture. Similarly, an angry expression in a context-free manner may be an indicator of inflexibility and self-interestedness, but in certain contexts it might also convey frustration or moral judgment, among others.

H1. *People's perceptions of an agent's trustworthiness are influenced by the action-consistency between the agent's emotional expression and its strategy.*

Agents whose expressions are consistent with their strategies will be preferred as partners for the trust game. Thus, when faced with a choice among self-interested or inflexible agents, people will tend to prefer angry ones; and when faced with a choice among non-self-interested or flexible agents, they will tend to prefer smiling ones.

H2. Within the negotiation, existing findings in the literature are expected to be confirmed; in particular:

(a) *The expression of anger will result in higher concession rates by humans*, as evidenced by Van Kleef et al. [130].

(b) *Agents who smile will help foster more integrative outcomes*, as evidenced by Carnevale and Pruitt [20]. Integrative outcomes are those in which the sum of the two players' payoffs is high. These can be achieved if the players realize there are mutually beneficial exchanges (such as one silver coin for

one iron coin) that increase both their payoffs.

(c) *Positive emotion will cause humans to also concede more points.* The theory of "emotion contagion" [56] predicts that people in negotiation games in which positive emotion is expressed will be more cooperative towards the agent.

4.2.3 Results

To assess people's behavior in the trust game, their choice of partner in the trust game was used as a metric. As was mentioned before, a trust game always came after three negotiation games, in which the agents differed in their emotion expression policy but not in their strategy. Therefore people were presented with a choice among three candidate emotion display policies (keeping strategy fixed). To investigate whether smiling agents or angry agents were preferred, it was tested whether $\Delta\mu = \mu_{smile} - \mu_{angry} = 0$ (in which μ_x denotes the fraction expression x was chosen), which would be the case if smiling and angry agents were equally preferred.[5] It is observed that, among selfish agents, $\Delta\mu = -0.25, p < 0.05$, denoting a preference toward angry agents. Among conceder agents, similarly, $\Delta\mu = -0.125$, although this is not statistically significant. A similar trend was seen with Nash agents ($\Delta\mu = -0.153$). On the other hand, among tit-for-tat agents preference is observed toward smiling agents ($\Delta\mu = +0.25, p < 0.05$). Notice that angry agents are being more preferred over smiling ones the more the strategy of the agent becomes inflexi-

[5]Agents implementing the fifth emotion expression strategy, "appraisal," were counted among the joyful or angry ones, depending on the emotion they actually expressed in the majority of rounds they negotiated with the subjects. The results are robust to including or excluding the "appraisal" agents' data.

ble or self-interested, confirming hypothesis H1 (results are summarized in Table 4.2). Mo effect of emotion was documented on the amount of resource trusted.

	Inflexible	**Flexible**
Non-self-interested	-0.153	$+0.25^{*}$
Self-interested	$-.0.25^{*}$	-0.125

Table 4.2: $\Delta\mu = \mu_{smile} - \mu_{angry}$, where μ_x denotes the fraction of times emotional expression x was preferred, for the various strategies in the trust game. Asterisk denotes that the mean is significantly different from zero.

To assess the influence of emotion expression on negotiation outcomes, both behavioral and subjective measures were used. Behavioral measures include variables relating to the negotiation game, such as the players' payoff at the end of the game, drop-out rates (i.e., the percentage of games with an agent in which humans dropped out, ending the game without a deal and awarding zero payoff to both participants), and measures indicating whether the games' integrative potential was exploited. Subjective measures, on the other hand, come from people's questionnaire responses after each negotiation game. Table 4.3 lists all the measures used. Let us now turn to the two hypotheses.

Surprisingly, anger was not observed having any effect of the average human concession rate (κ), thus disconfirming hypothesis H2(a). Figure 4.3 plots the average concession rate across emotion expressions for all four strategies examined. As can be seen, the average concession rate of the people was strongly influenced by the strategy of the agent, but very little by its emotion expression. Similarly, no support was found that the drop-out rate (d), or the integrative potential, as measured by π_{Σ} or π_{Π}, was influenced by the choice of emotion across all strategies (thus H2(b) is

also rejected). Finally, the hypothesis that positive emotion will influence concession rates according to the theory of "emotion contagion," according to H2(c), was also not supported by the measurements.

Behavioral Measures
π^a : agent points at the end of the game
π^h : human points at the end of the game
$\pi_\Sigma = \pi^a + \pi^h$: sum of payoffs
$\pi_\Pi = \pi^a \pi^h$: product of payoffs
κ : average human concession rate between two rounds
d : average drop-out rate (%)
Subjective Measures (Likert 1-7)
q_1 : how much did you like this agent?
q_2 : how much did you feel this agent cared about you?
q_3 : how human-like was the agent?
q_4 : would you play with this agent again?

Table 4.3: Behavioral and subjective measures.

It must be noted that action-consistency seems to also play a role in people's subjective reports. With respect to "liking" (q_1) people showed an aversion towards always-angry agents for flexible strategies, but not for inflexible ones. With respect to how much people felt the agent cared about them (q_2), always-smile agents were preferred under non-self-interested strategies. Also, with respect to people's expressed desire to play with an agent again in the future (q_4), an aversion was seen toward always-angry agents only among agents playing the tit-for-tat strategy. All the above results indicate that action-consistent expressions are preferred over others. Finally, looking at how much the agent was perceived to be human-like (q_3) no effects of emotion were noticed.

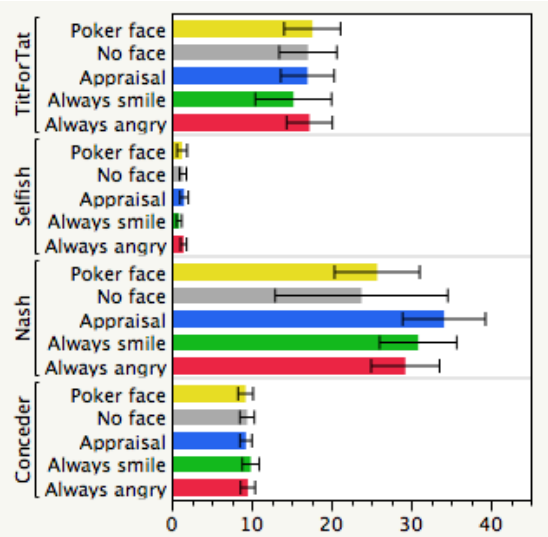

Figure 4.3: Average concession rate (κ) per emotion expression and strategy in the negotiation game (error bars denote 95% confidence intervals).

Supplementary results

This section presents further findings of the experiment that were not directly related to the hypotheses examined. These are reported for two reasons: (a) because they might be useful in comparing with other negotiation experiments in the literature, and (b) as evidence that the various strategies the agents employed did make a difference in people's behavior (in other words, the strategy choice was not superfluous). Hence Figure 4.4 displays the points in the possession of the agent and the human at the end of the game. Here it can be seen that the conceder and selfish agents (self-interested) fare better in terms of final score than the non-self-interested agents. Also, the Nash agent causes very equitable outcomes to be obtained.

Figure 4.5 show the effect of the selfish agents on humans who choose to drop-out, hence punishing the agent at a cost (they both receive zero points). As can be seen

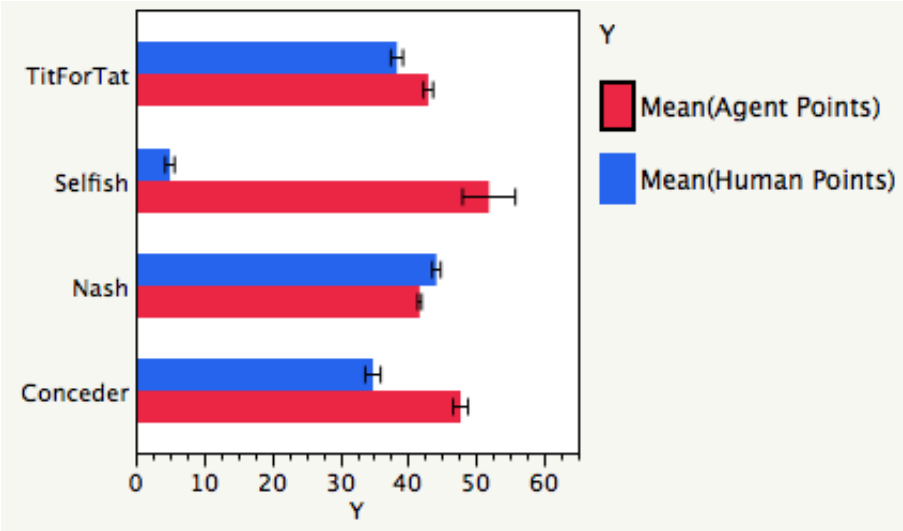

Figure 4.4: Average human and agent points in the end of the game per strategy (error bars denote 95% confidence intervals).

in the chart, no player drops out against any other agent, but up to a quarter of the participants do when their partner shows such intransigence and self-interestedness.

Finally, Figure 4.6 shows the average duration of the game in rounds. It is clear that people agree to a deal sooner with non-self-interested agents, but try harder against self-interested ones. (Interestingly enough, a small but statistically significant difference of emotion expression on the game's duration in the case of the selfish agent was observed. In that subset of the data, it can be seen that smiling causes people to play on average for two more rounds with the selfish agent, trying to forge a deal, before they accept its terms or drop out. This is perhaps because smiling conveys an interest—on the agent's behalf—to be cooperative and work together with the human, which keeps her trying for longer before conceding to the fact that the agent will not change its mind.)

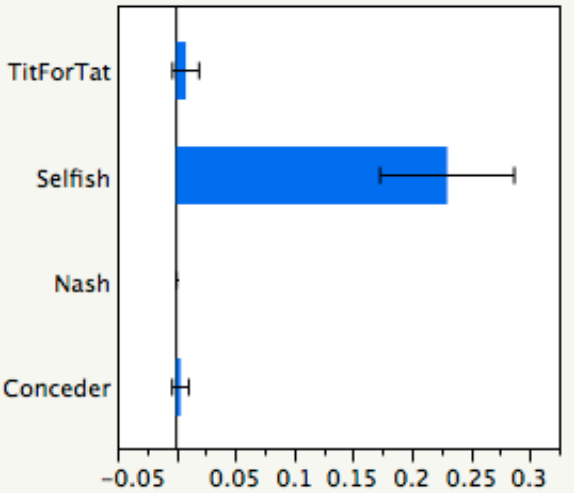

Figure 4.5: Average human drop-out rate per strategy (error bars denote 95% confidence intervals).

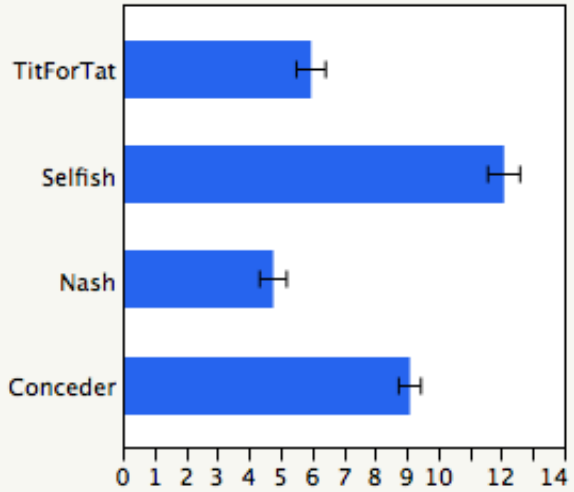

Figure 4.6: Average duration of game in rounds per strategy (error bars denote 95% confidence intervals).

4.3 Discussion and Extensions

The study discussed above demonstrates the role of emotion expressions in human-computer interaction. Agents were perceived as more trustworthy when they deployed appropriate emotion expressions that were consistent with their actions. The study reveals important ways in which emotion expressions, and non-verbal cues in general, can influence people's perceptions and behavior.

The reason why people are influenced has been a subject of debate among researchers. Kahneman and Tversky [129] back in the 1970s have argued against the notion that people are always rational. They claimed that people employ a set of heuristics and often exhibit consistent biases in their reasoning and decision-making. As an example, the *availability* heuristic is the phenomenon n which people predict the frequency of an event, or a proportion within a population, based on how easily an example can be brought to mind. Thus someone might argue that cigarette smoking is not unhealthy because his grandfather smoked three packs of cigarettes a day and lived to be a hundred. The grandfather's health could simply be an unusual case that does not speak to the health of smokers in general, but because it is "available" in the person's memory, it biases his opinion.

Later Kahneman [62] and other economists [124] have suggested the *two-system hypothesis*, according to which human decision-making is performed by means of two systems: System 1 (S1) is fast, frugal, "quick and dirty" and leverages heuristics, emotions, quick associations and conditioned reactions to make decisions quickly, but not always accurately. System 2 (S2), on the other hand, is slower and deliberative. S2 costs more to operate but is more accurate, while S1 is faster but often wrong. In

the picture these researchers have drawn, human rationality emanates from S2, while biases and errors arise out of S1. This idea has received criticism from some neuroscientists as of late, as brain regions typically associated with S1 (e.g., the amygdala, one of the emotion centers of the brain [37]) and those associated with S2 (e.g., the prefrontal cortex, responsible for predicting the consequences of actions, among other activities) have been shown to be less distinguishable and more interconnected than previously thought. As such, the idea that two distinct systems govern thought and decision-making has come into question.

Besides these neuroscience-related objections, the two-system hypothesis has also come under attack from evolutionary psychologists. Although it relaxes the notion of absolute rationality by allowing for a "locus of biases" (S1), the theory still treats such phenomena as emotion-influenced decision-making as disturbances of solid and correct reasoning. Mood influences, cognitive heuristics and departures from rationality are attributed to design imperfections in the brain or due to its limited computational capacity, but no positive role is recognized to them in the two-system hypothesis.

On the contrary, evolutionary psychologists have argued that the emotions, much like any aspect of our bodily or psychological design, have evolved to address frequently-encountered problem in our evolutionary past. According to Cosmides and Tooby [26] emotions are *microprograms*, short behavioral scripts, each with an activation condition that prompts its execution under the right set of circumstances. These microprograms governed the individuals' behavior and were one of the determinants of their procreative fitness. Hence, populations with more *adaptive* microprograms prospered under natural selection. As such, emotions are not at all irrational, but

highly useful means by which decisions can be reached in environments that resemble those ancestral domains. Of course, a corollary of this argument is that emotions might be *maladaptive* in situations that depart in structure from these environments that helped promote them in our species' evolution.

The influence of emotions—and non-verbal cues in general—on human perception and behavior can then be interpreted under this light. People's response to a certain cue can be expected to be modulated by the function of that cue in frequently encountered settings.[6] In most studies demonstrating that people treat computers as social actors the effects recorded can be attributed to the way signals and behaviors are interpreted and processed by these microprograms. Recall from the introduction, for instance, that reporting a happy story with a happy voice makes people perceive it as more positive. This is simply a manifestation of the fact that affective signals like a happy voice carry meaning in life. A person reporting a story in a happy voice is assumed to be in a state of elation, whose cause is then reflected to the characteristics of story being reported. Taking the non-verbal cue out of context does in no way stop the relevant cognitive processing functions from being activated.

Emotion expressions are also a frugal way to communicate information, as reported in the last chapter. Their role in human-computer interactions studies can then be understood in this context as well. If an emotion expression is a simple signal that communicates a wide breadth of information, placing that signal in an interaction

[6]These frequently-encountered settings are not necessarily limited to frequent situations in evolutionary history, but can be extended to frequent situations in the person's lifetime, or her culture. As a matter of fact, one of the adaptive functions of emotions is that they can have their activation conditions *re-purposed*, so that they be invoked by novel situations beyond those genetically prescribed [96].

serves to economically communicate all this information. One can imagine such "low-bandwidth" signals being used constructively to influence not just people's subjective perceptions of computer systems, but also influence their behavior towards those systems in constructive ways.

Similarly, the emotion expressions in the study serve to reinforce or contradict the personality characteristics that the agent's actions already convey. If the agent is a tough negotiator but expresses positive emotions people get conflicting signals. On the one hand, the agent through its actions shows little regard toward their well-being; on the other hand, through its non-verbal expressions it advertises a willingness to work together peaceably. The resulting confusion makes the agent less predictable, and hence less likely to be trusted in the future in a situation of uncertainty, such as the trust game in the study. If, however, the agent's emotion expressions and actions are consistent they both reinforce the same characteristics, increasing the agent's predictability, and hence its perceived trustworthiness.

Chapter 5

The Reasoning Patterns: Simplifying and Representing Games

Strategic environments are another place in which increased complexity makes reasoning and decision-making more difficult for agents. Game theory offers a principled way to analyze and predict the behavior of rational agents in such multi-agent strategic environments. Section 1.2.2 of the Introduction offers a short description of the underlying assumptions, most important concepts and implications of game theory. In brief, the theory assumes that all agents possess a utility function that they are trying to maximize by formulating appropriate strategies. Agents are expected to best-respond to each others' strategies in what is essentially a fixed point in the strategy space, referred to as a Nash equilibrium of the game.

Game theory is useful for the design of agents, as it provides a way for them

to predict and anticipate the strategic behavior of other agents, and thus to select appropriate best-responding strategies to maximize their own utility. Game-theoretic models are also useful for reasoning about the game as a whole in three different ways. First, they can be used for descriptive analysis: an outside *modeler* of the game can use them to understand why agents behave a certain way. Second, they can be used for prescriptive analysis: a modeler can use them to build agents that implement strategies outperforming those of existing agents. Third, they can be used for mechanism design: the modeler can alter the design of the game to induce agents to adopt particular desirable behaviors.

To analyze a game, a modeler must reason about the interactions among the agents in the game, the consequences of their actions, the flow of information among them, their beliefs and observations, and their strategic thinking. Understanding the reasons why agents choose a particular strategy is helpful in descriptive analyses, as it offers an intuitive understanding of their reasoning. It also enables easier prediction of how these agents will behave under different circumstances or in different environments, thus making it useful in prescriptive analyses and mechanism design.

Standard game-theoretic accounts offer a very coarse understanding of why agents choose their actions. They assume that the agents' only objective is to maximize their utilities. In complex games, however, directly maximizing an agent's utility might be computationally hard. In such games it might be useful to reason in a finer way about the agents' objectives. In particular, the structure of the game can reveal specific ways in which agents can achieve high utility. Consider an example, in which an agent's actions are observable by other agents. In this case, the agent might benefit

by considering how these agents' observations will influence their beliefs and future actions. Being able to anticipate their responses, it can steer them towards choices that improve its utility. Notice that this line of reasoning about a game says more about agents' reasoning and objectives. Agents' choices are examined not only with respect to the utility they provide, but also with respect to the dynamics within the game that result to it.

Pfeffer & Gal's theory of reasoning patterns [103] offers a principled account of these kinds of strategic reasoning. In this chapter, I show how to use this theory to assist game modelers, especially in complex games, while the next chapter presents an application of the theory to assisting human decision-makers. Section 5.1 describes the reasoning patterns and the ways they capture the flow of utility and information within a game. Four types of reasoning pattern types suffice to describe the objectives of every player in every game, and these four types of reasoning patterns can be interpreted in relatively straightforward, intuitive ways. The theory of reasoning patterns is used to make two distinct contributions.

First, as discussed in Section 5.2, the theory of reasoning patterns is extended to games of incomplete information (Bayesian games), with and without a common prior. For this to be accomplished, a novel graphical formalism was introduced for such games. This new formalism allows agents' beliefs, no matter how divergent, to be represented in a concise *belief graph*, which enables us to define reasoning patterns across the belief sets of two or more agents, and to identify them easily, even when these beliefs sets are mutually inconsistent.

Second, this chapter describes a technical, game-theoretic contribution. In Section

5.3, it is demonstrated that the reasoning patterns can be used to simplify certain games for the purpose of computing a Nash equilibrium, by recognizing decisions in those games that can be ignored during the equilibrium computation. In some classes of games this may lead to exponential time savings. A polynomial-time algorithm for the efficient detection of reasoning patterns in the MAID representation of a game is also presented.

5.1 Reasoning Patterns in Games

Gal & Pfeffer [103] investigate the reasons why agents choose their strategies in a game. The fundamental objective of every player is assumed to be, consistently with every other game-theoretic model, achieving the highest possible utility for itself. But the way in which players may act to achieve high utility depends critically on the structure of the game and the nature of their interactions with other players. For instance, a player may choose to behave in a way that reveals private information it possesses to others, or it may act to manipulate their beliefs to its own benefit, if the game's structure enables such strategic acts. The theory identifies systematic patterns behind agents' strategic reasoning, termed the "reasoning patterns."

As Multi-Agent Influence Diagrams will be extensively used in presenting and discussing the four types of reasoning patterns, we begin our discussion with a short introduction of their semantics and most important concepts.

5.1.1 Multi-Agent Influence Diagrams

A Multi-Agent Influence Diagram (or MAID [71]) is a concise graphical representation for games, like the Normal or Extensive forms (see Section 1.2.2). A MAID is a graph that shares many of the structural features of a Bayesian network [102]. It consists of nodes, which might be chance nodes, rectangular nodes, or utility nodes, as well as directed edges connecting these nodes. For each node X we will denote its parents in the MAID graph as $Pa(X)$.

Chance nodes are ellipsis-shaped and correspond each to a random variable that takes values stochastically. In particular, every chance node C has a *domain* $A(C)$, which is a set of possible values for that variable $A(C) = \{c_1, \ldots, c_{|A(C)|}\}$. Notationally, we will use capital letters to refer to random variables or the corresponding chance nodes, and lowercase letters to refer to particular values for that variable. If a chance node has no parents in the MAID graph ($Pa(C) = \emptyset$), then it is associated with a probability distribution over its domain, called the "prior" for that node, $p(C) = \{p^C_1, \ldots, p^C_{|A(C)|}\}$, such that $p^C_i \geq 0, \forall i$ and $\sum_i p^C_i = 1$. If the chance node has parents in the graph, then instead of a prior it is associated with a "conditional" distribution $p(C|Pa(C))$, which stochastically selects values for that variable among its domain, based on the values of its parents. In this way, chances nodes have the exact same semantics as nodes in a Bayesian network.

Decision nodes are rectangular in shape and represent points along the game when agents take actions. Each decision node D is associated with an agent that takes the action in that node, denoted $P(D)$. The domain of the node, $A(D)$, is the set of actions available to the agent. Finally, the parents of a decision node denote what

the observations of the agent are when it is making its decision. For instance, if decision node D has chance node C as one of its parents, this means that agent $P(D)$ observes the value of variable C when making its decision. Similarly, if decision node D has decision node D' as one of its parents, this is interpreted as agent $P(D)$ having observed the action selected by agent $P(D')$ in D'.

Utility nodes, finally, are diamond-shaped and represent agents' payoffs. A utility function U is associated with a particular agent $P(U)$, whose utility it represents. Its parents denote the set of variables (chance or decision nodes), whose value influences the agent's utility. In particular, each utility node U is associated with a deterministic (not stochastic) function that maps the values of its parent nodes to a real number ($U : \times_{X \in Pa(U)} A(X) \to \mathbb{R}$). If a MAID contains more than one decision node for an agent i, the agent's total utility is taken to be the sum of the values of all these nodes: $u_i = \sum_{U:P(U)=i} U$.

A strategy for an agent i in a MAID consists of a "decision rule" $\delta(D)$ for every decision node D belonging to that agent ($P(D) = i$). This decision rule specifies how the agent is going to decide an action within the set $A(D)$ in decision D, by looking at the observations the agent has at the time of its decision, i.e., the values of the parents $Pa(D)$ of D. Formally, $\delta(D)$ maps the values of $Pa(D)$ to a probability distribution over $A(D)$, allowing for stochastic decision-making.

Consider an example MAID with just a single agent to make the above definitions clearer. A person must decide whether to take an umbrella with her in the morning (decision D). She certainly wants to be carrying an umbrella in case it rains, but she prefers not to carry the extra weight of the umbrella on a sunny day. Let us

assume that the only observations she has are: (a) the weather forecast (F), and (b) yesterday's closing prices of stocks (S). Another variable, the true weather (W) is unobservable by the decision-maker (hence not a parent of D), but the forecast reveals information about that hidden variable indirectly. In other words, F probabilistically depends on W; for example, on rainy days the forecast is also likely to be predicting rain, but it might erroneously predict good weather sometimes. The person's actual utility depends on her choice D (take umbrella or not) and the true weather W (rain or sun). The stock market prices neither reveal anything about the weather, nor do they influence her utility in the context of carrying an umbrella. Figure 5.1 presents the MAID for this simple game.

In this MAID, chance nodes S and W are each associated with a prior distribution. For instance, W might be 'rainy' with probability 0.2 and 'sunny' with probability 0.8. Node F, since it has a parent, has a conditional distribution $p(F|W)$. For instance, F might be 'rainy' with probability 0.75 when W also indicates rain; when W is 'sunny,' F might be 'rainy' with probability only 0.15. Decision node D might have a domain set containing actions 'take umbrella' and 'not take umbrella.' Finally, the utility of the agent is $+10$ on a sunny day on which the agent does not carry an umbrella, $+2$ on a sunny day with an umbrella, 0 on a rainy day with an umbrella, and -20 on a rainy day without an umbrella. Notice that, since S is not a parent of U, the agent's utility does not depend on the values of S.

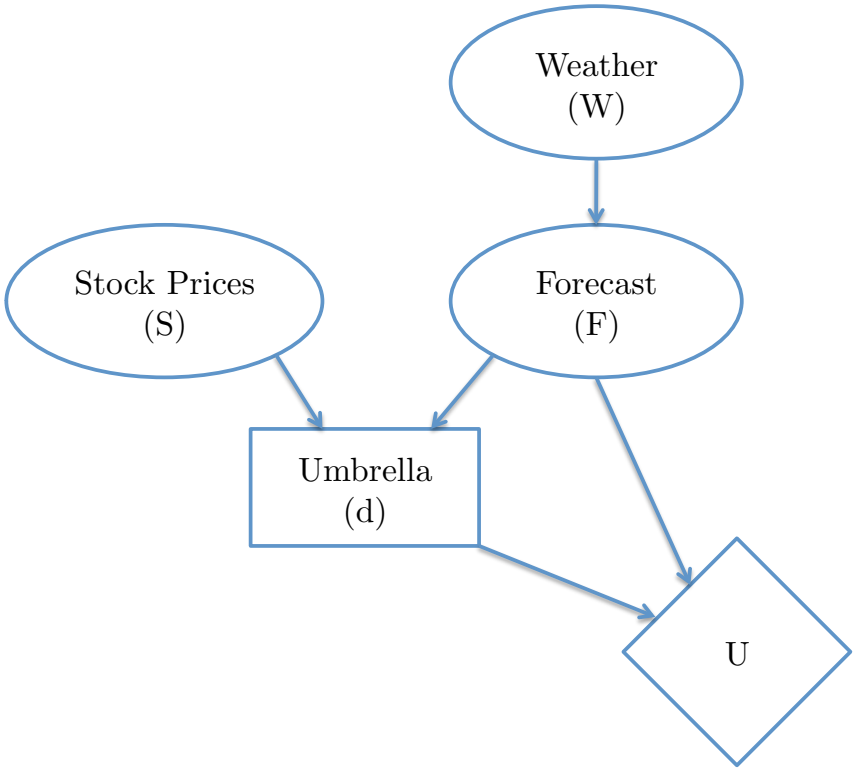

Figure 5.1: The MAID for the simple example game

Relationship between MAIDs and other formalisms

A MAID is a formalism to represent games, and as such it shares similarities with the Normal and Extensive forms, described in Section 1.2.2. That relationship is not straightforward, however. To derive the Normal form representation of a game for its made, we can proceed as follows: For every decision node D of each player i, consider every possible combination between the possible values of its parents $Pa(D)$ and possible actions in the domain of the decision node $A(D)$. The sum of these combinations is the set of pure strategies S_i for player i. The payoff to each player i when players follow strategy profile $s = (s_1, \ldots, s_n)$ is the sum of values of i's utility

nodes, when every player j uses the decision rules specified by its strategy s_j.

To construct the corresponding Extensive form game tree, the process is similar. Every chance node can be represented as an information set ruled by the 'Nature' player, in which the probabilities associated with each action are given by the corresponding prior or conditional distribution of each node. For every decision node D of player i, also add an information set for all possible values of its parents $Pa(D)$. The leaf nodes of the tree will be assigned utility values computed in the same way as for the Normal form representation.

MAIDs are also closely linked to Bayesian networks. One can transform a MAID into a Bayesian network by fixing the strategies of all players in it. Since strategies offer decision rules for each decision node D mapping $Pa(D)$ onto probability distributions over the domain $A(D)$, we can replace the decision node with a chance node implementing the decision rule as a conditional probability distribution. We can do something similar for utility nodes, replacing them with chance variables associated with a deterministic conditional probability distribution over the set of real numbers. The resulting structure is a Bayesian network that captures probabilistic dependencies between the random variables, the agents' actions, and their utilities.

Well-Distinguishing (WD) strategies

Before the four reasoning patterns types are presented, a key technical assumption needs to be mentioned. Pfeffer & Gal assume that all players adopt *well-distinguishing* (WD) *strategies*. Intuitively, WD strategies do not condition an agent's decision on irrelevant observations. Consider the previous example in Figure 5.1. The observa-

tions of the decision-maker in this game are F and W, the forecast and the stock market prices. A strategy for her is then a mapping from stock market prices and forecasts to distributions over her action set, containing choices 'take umbrella' and 'not take umbrella.' For instance, a possible strategy might be "if the stock market went up more than 5.4% and the forecast predicts rain, then take an umbrella with probability 30%; otherwise, take an umbrella with probability 2%." The above strategy is not WD. Well-distinguishing strategies would never condition the player's decision on stock-market prices, as they neither influence the agent's utility, directly or indirectly, nor do they reveal anything about any variable that does.

Formally speaking, consider the observation set $Pa(D)$, an observed variable $X \in Pa(d)$, and define set $V^X(D) = Pa(D) - X$, the set of all observations excluding X. Also, consider $\hat{u}(pa(D), \mathbf{Do}(a))$ to be the expected utility for the agent making a decision in D upon choosing action $a \in A(D)$ with assignment of values $pa(D)$ to observations $Pa(d)$.[1] For every observation variable $X \in Pa(d)$, for every assignment of values x_1, x_2 to variable X, if for every assignment of values $v^X(D)$ to all other variables except X it holds that $\hat{u}((o_1, v^X(D)), \mathbf{Do}(a)) = \hat{u}((o_2, v^X(D)), \mathbf{Do}(a))$ for every decision $a \in A(D)$ of the agent, then a well-distinguishing strategy should not condition on variable X, but instead be a mapping from $V^X(D)$ onto probability distributions over $A(d)$.

Back to our example, consider the observation S, the stock market prices. For every possible assignment to the forecast variable F (sun or rain), for every action the agent takes (take umbrella or not), her expected utility does not change based

[1]Consistent with our notation, we use the uppercase to refer to the observed variables, $Pa(D)$, and the lowercase to refer to their their specific values, $pa(D)$.

on the stock market prices. Hence, should never condition on S if her strategy is to be well-distinguishing. A WD strategy for the player would be, for instance, "if the forecast predicts rain, then take an umbrella with probability 40%; otherwise, take an umbrella with probability 4%."

Definitions aside, is it plausible to assume that players only use well-distinguishing strategies? For rational, utility-maximizing agents, it easy easy to show that for every non-well-distinguishing strategy they may implement, they can obtain the same utility using a well-distinguishing strategy.[2] For human players, empirical studies have revealed that they do *not* always follow well-distinguishing strategies. At this point, however, and for the remainder of the chapter, we are going to assume that players limit their strategic choices among well-distinguishing strategies. Admitting non-WD strategies into the analysis of a game would allow for a very large number of "bizarre" behaviors to be considered. For the purpose of illustration, consider two examples. It is non-WD in poker to always 'raise' if the player to your right has brown hair, although his hair color does not influence your own utility. It is non-WD to sell a stock if its ticker starts with the letter 'S' or the day of the week is Tuesday, although this information is irrelevant to the stock's prospects. Most would clearly consider such behaviors to be non-sensical. Excluding non-WD strategies makes the analysis of agents' objectives significantly more concrete, so we will henceforth assume strictly WD strategies by all players.

[2]That said, the new well-distinguishing strategy might no longer be in equilibrium with the strategies of other agents. The set of equilibria in which all agents use well-distinguishing strategies is a (possibly proper) subset of the full set of equilibria in every game.

Blocking Paths

In presenting the reasoning patterns, we shall use an important technical concepts, the "blocked path." As mentioned above, a MAID is a directed graph containing chance, decision and utility nodes, and we can treat a MAID as a Bayesian network by fixing the strategies of agents. We can then define a *path* on a MAID (or Bayesian network) as a sequence of nodes connected by arcs, like on any other graphical structure. Paths can be directed (e.g., $< n_1 \rightarrow n_2 \rightarrow \ldots \rightarrow n_m >$) or undirected (e.g., $< n_1 \leftarrow n_2 \rightarrow \ldots \leftarrow n_m >$), but we assume that they do not contain circles. A path must contain at least two nodes. Nodes along the path have converging arrows, diverging arrows, or sequential arrows.

- A node n along a path π is said to have *converging arrows* if it is not the first or the last node of the path, and the both edges along the path are incoming to it. For instance, in path $< n_1 \rightarrow n_2 \leftarrow n_3 >$, the node n_2 has converging arrows.

- A node n along a path π has diverging arrows if it is not the first node in of the path and both edges along the path are outgoing from it, for instance nodes n_2 and n_4 in path $< n_1 \rightarrow n_2 \rightarrow n_3 \leftarrow n_4 >$.

- Finally, a node has sequential arrows, if it has neither converging or diverging arrows, e.g., node n_2 in path $< n_1 \rightarrow n_2 \rightarrow n_3 >$ or path $< n_1 \leftarrow n_2 \leftarrow n_3 >$.

Regarding blocked paths, we shall define path π to be *blocked* by the set of nodes Z, called the "blocking set," at node $n \in \pi$, called the "blocking node," when either

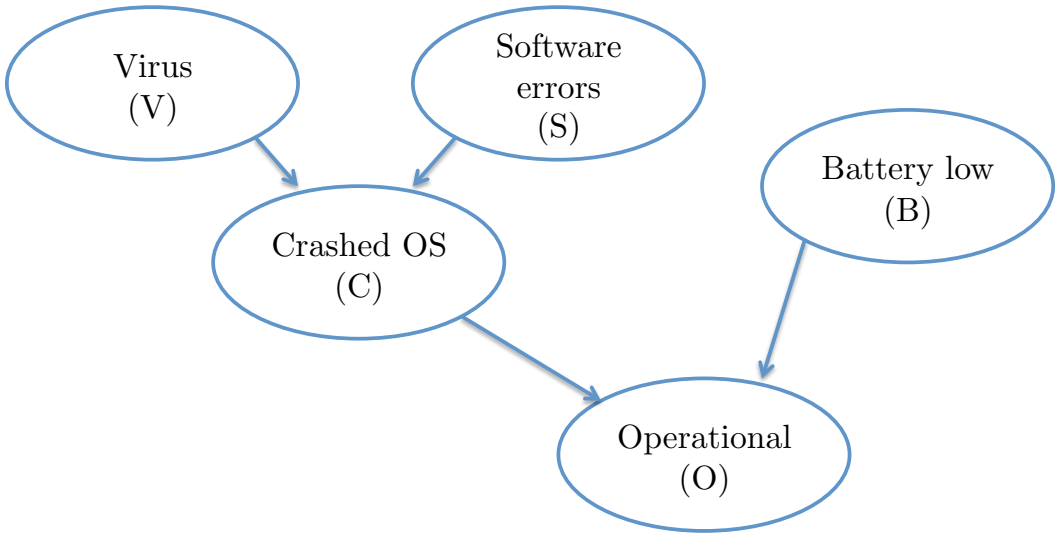

Figure 5.2: A BN example illustrating blocking

of the following two conditions holds: Either n does *not* have converging arrows and $n \in Z$, or n has converging arrows and neither n nor any of its descendants are in Z.

To understand intuitively the concept of blocking, consider a simple example: a robot may be operational or not, which is represented as binary variable O. There are two reasons why a robot might be non-operational: either it is out of battery (binary variable B) or its operating system has crashed (variable C). Operating system crashes may be caused by a virus (V) or software errors (S). The Bayesian network for this simple scenario is shown in Figure 5.2.

Blocking paths help illustrate the "flow of inferences." The blocking set Z can be interpreted as the set of variables whose values are observed (known). A path between two unobserved variables X_1 and X_2, when not blocked by blocking set Z, is then interpreted as variable X_1 being able to change our beliefs about variable X_2,

despite us knowing the values of variables in Z.

Suppose, for instance, that we have observed the system having a virus, that is $Z = \{V\}$. Consider path $\pi =< V \rightarrow C \rightarrow O >$. This path is not blocked in any of its nodes by set Z, because variable C does not have converging arrows and $C \notin Z$, and the same is true for variable O. This allows inference to "flow." In other words, knowing that there is a virus makes it more likely that the operating system has crashed, which makes it more likely that the robot is not operational. Evidence from variable V allows us to make inferences about all the other variables downwards along the path. Notice that the same is true if we observed that the robot was non-operational instead, i.e., $Z = \{O\}$. Again, path π is not blocked and inference can similarly "flow." The fact that the robot is non-operational would make it more likely that there is an OS crash, and by extension more likely that the system has a virus.

Consider another case, now, in which the inference path is blocked. Suppose we observe that the operating system has crashed, that is, $Z = \{C\}$. Take again path $\pi =< V \rightarrow C \rightarrow O >$. This time, the inference does not "flow" through variable C. To see this, consider the effect of a virus in the system (V) on whether the robot is operational or not (O). Since we already know that the operating system has crashed, the presence of a virus can tell us nothing more about the operational status of the robot. In other words, there is no influence of the virus's presence on the operational status of the robot, except the one that "goes through" the crashing of the operating system. Since we already observe this ($C \in Z$), there is nothing more that we can learn by reasoning about the virus. Hence inference from V to O is blocked at C.

For an example with converging arrows, suppose we observe that the system is not operational ($Z = \{O\}$). This of course makes it more likely that the battery is low, along path $< O \leftarrow B >$. It also makes it more likely that the OS has crashed, along path $< C \rightarrow O >$. But consider now path $\pi' =< C \rightarrow O \leftarrow B >$, on which node O has converging arrows. Having observed that the system is non-operational, we can still perform inferences from variable B to variable C. In particular, if the battery is low, the probability that the system has crashed now decreases. This is because the battery being low can explain the non-operational status of the robot, which makes it less likely that an OS crash is also responsible for it. Because $O \in Z$ and O has diverging arrows along path π', the path is not blocked and such inferences are able to "flow." If, instead, we observed only the battery status ($Z = \{B\}$), then we would not be able to say anything about an OS crash. A low battery and an OS crash are independent events, and observing one says nothing about the likelihood of the other, without also observing the robot's operational status. In that case, path π' is blocked at O, since it has converging arrows and $O \notin Z$.

The Reduced MAID

In MAID graphical notation, as discussed above, any observations available to an agent i when making a decision D must be parents of node D in the MAID graph. Hence the decision rule for D is essentially a mapping from the parents of D to a probability distribution over the actions $A(D)$. If agents follow WD strategies they cannot, however, condition their choices on irrelevant variables. There is a simple graphical way to discern which variables among $Pa(d)$ are irrelevant for the agent

making a choice in D. A variable $X \in Pa(D)$ is irrelevant if all paths from X to any utility node U of agent i are blocked by set $Pa(D) - X$. Extending this definition to more than one variable, a subset $K \subseteq Pa(D)$ is irrelevant if all paths from any node in K to any utility node U of agent i are blocked by set $Pa(D) - K$.

As mentioned before, a variable X is irrelevant when it has no impact on the agent's utility. Intuitively, then, a variable "matters" (is not irrelevant) for a decision when its value reveals important information about what the right decision should be. In other words, a variable matters when inference can "flow" from that variable to a utility node of the agent in the MAID graph, i.e., there is a path from that variable X to some utility node U of i. However, that inference should not be mediated exclusively by all the other observed variables $Pa(D) - X$. If that were the case, then X offers no more insight than the remaining variables, hence it is de facto irrelevant. Therefore, if all paths between X and all U of agent i are blocked by $Pa(D) - X$, then X is irrelevant.

To reflect that all players are required to follow WD strategies in a game, we can take every decision node D in the MAID and remove the edges connecting it to any of its irrelevant parents. This process can be repeated iteratively, as severing an edge might cause other paths to become blocked and more variables to be marked irrelevant. The final output of this process is called the *reduced MAID*. In the discussion of the reasoning patterns that follows, we shall assume that all MAIDs are in their corresponding reduced form.

Conditional Blocking Set

Finally, we shall introduce one more notational element. We shall use W_X^Y, called the conditional blocking set of node X for node Y, to denote all parents of X that are not descendants of Y in the maid graph. Formally, $W_X^Y = Pa(X) - \{n : n \in Pa(X) \wedge \nexists \pi =< Y \rightarrow \ldots \rightarrow n >\}$. The conditional blocking set will be important in the definition of the reasoning patterns, discussed below.

The Four Types of Reasoning Patterns

Pfeffer & Gal identify four types of reasoning patterns, presented below. For each pattern an intuitive description is offered, followed by a formal definition and a correspondence to graphical properties of the MAID representation of the game.

These four reasoning pattern types are important because they are "complete," in the sense that they can fully characterize the objectives of a rational agent under WD strategies. Pfeffer & Gal prove this in a theorem. They denote a decision D of agent i *motivated* if agent i has a reason to deliberate about his choice in D. Technically speaking, decision D is motivated if there is an assignment $pa(D)$ to its parents $Pa(D)$, and two actions $a_1, a_2 \in A(D)$, such that $\hat{u}_i(pa(D), Do(a_1)) > \hat{u}_i(pa(D), Do(a_2))$, for some strategy choice of all other players except i. In that case, the agent has a reason to deliberate, because choosing action a_1 yields him a higher expected utility over action a_2. (If all actions yielded the same expected utility under all possible observations available to agent i in D, then his choice would have no effect whatsoever, hence the decision would not be "motivated.") Pfeffer & Gal prove that, if a decision D is motivated, then it must be associated with at least one of the

four types of reasoning patterns described above. As a consequence, a decision node that is associated with no reasoning pattern is by definition unmotivated. They also prove a weaker version of the converse: If a decision node has one or more reasoning patterns in a MAID graph, there exists an assignment to the MAID parameters that will make this decision motivated.[3]

1. The first reasoning pattern is *direct effect*. It refers to instances in which a player can influence his utility without the intervention of another player being required. This influence could be deterministic or stochastic. Consider an example: Andrew and Beatrice, with utility functions u_A and u_B, respectively, are playing the 'matching pennies' game, in which both simultaneously choose either Heads or Tails; if they select the same choice, then both gain \$1, otherwise, they both lose \$1. In this game Andrew has direct effect, since his choice impacts his utility, causing it to be $u_A(1)$ or $u_A(-1)$ for a given choice by Beatrice. Notice that Andrew's utility depends *also* on Beatrice's choice, not just his own. Direct effect only requires that Andrew may unilaterally influence his utility by his choice, not that this ought to be the only source of influence. Beatrice's decision in this game has direct effect for the same reasons. The MAID for this game is presented in Figure 5.3(a).

   In the previous game agents influenced their utilities deterministically by their choices. Consider now an extension of this game. In the beginning of a game players select Heads or Tails, as in the basic version, but then an actual coin is

[3]This converse theorem is weaker because the presence of a reasoning pattern might, under some extreme choices for MAID parameters, cause a decision to remain unmotivated, c.f. [103], p.6.

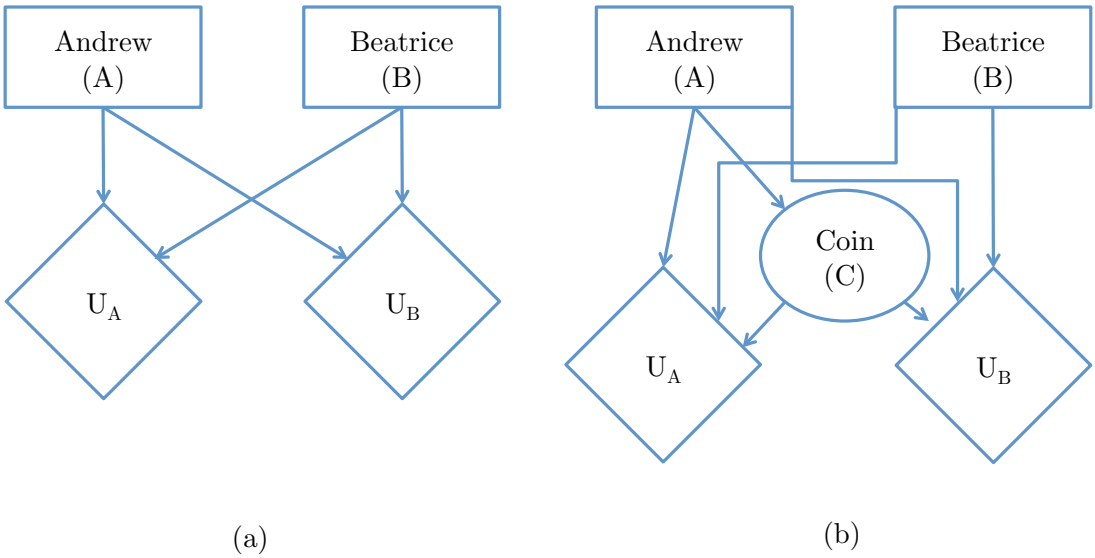

Figure 5.3: Two examples of direct effect

tossed. There are two choices for that coin: if Andrew has selected Heads, coin 1 is tossed, which is a fair coin; otherwise, if he has selected Tails, coin 2 is tossed, which comes up Heads with only 10% probability. Again, players win or lose one dollar depending on whether their own choices match or differ, respectively. The only difference is that, if the coin lands on Heads, their wins or losses are doubled. Notice now that players' utilities are not deterministic but can be computed in expectation only. Despite the stochasticity introduced, however, both agents' decision still have direct effect. The MAID for this extended version of game is presented in Figure 5.3(b).

In the MAID representation of a game, decision D of agent i has direct effect if there is at least one directed path from the decision node D to a utility node U_i of agent i that does not contain any decision nodes of other players $j \neq i$.

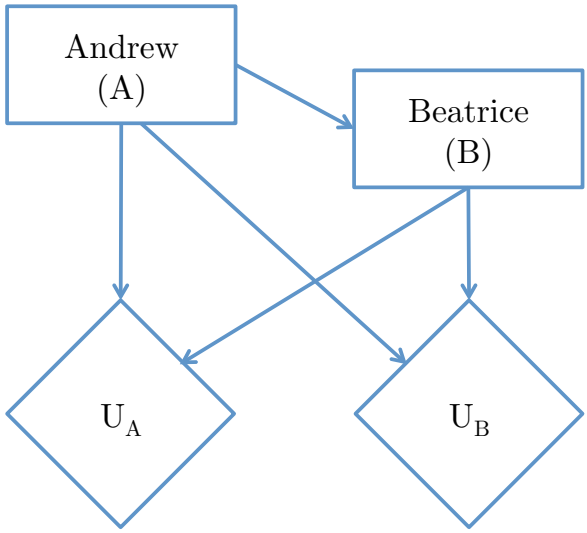

Figure 5.4: An example of manipulation

The path might, however, contain chance nodes, if the influence is stochastic, like in the second version of our game.

2. The second reasoning pattern type is *manipulation*. This reasoning pattern refers to cases in which another player is involved, and her decision can be influenced (manipulated) in desirable ways. Consider again a simple example: Andrew and Beatrice play the matching pennies game as before, but this time Andrew announces his choice of Heads or Tails, then Beatrice makes her choice after having observed Andrew's announcement. The MAID for this game is presented in Figure 5.4.

   In this game Andrew can influence Beatrice's choice by means of his announcement. If he selects Heads, for example, Beatrice will also wish to select Heads, obtaining a utility of $u_B(1)$ for herself, as opposed to $u_B(-1)$ if she instead

selected Tails. Three factors make this influence mechanism work: First, Andrew's decision is observable by Beatrice (arrow from decision node A to node B in the MAID). In the standard game in which both players make decisions simultaneously and independently, Andrew could never hope to influence Beatrice, as his choice cannot be known by her at the time of her decision. Second, Beatrice's decision influences Andrew's utility (arrow from B to u_A). If that were not the case, Andrew would have no reason to bother influencing her choice. And third, Andrew's choice impacts Beatrice's utility (arrow from A to u_B). If that were not true, then Beatrice would never alter her decision based on Andrew's action. (Remember that we only allow for well-distinguishing strategies. If Beatrice could adopt non-WD strategies, she could arbitrarily condition her choice on Andrew's, whether his actions mattered for her utility or not.)

The formal graphical definition of manipulation relies on these two conditions, broadly considered. There is thus manipulation for decision D of agent i if: (a) there is a directed, decision-node-free path from D to a decision node D' of another agent $j \neq i$; (b) there is a directed path from D' to a utility node U_i of agent i; and (c) there is a directed path from D to a utility node U_j of agent j that does not pass through D'. The paths in (b) and (c) may contain decisions of agent i, agent j, or some other agent.

3. The third reasoning pattern type is *signaling*. This reasoning pattern, like manipulation, involves influencing another player. But whereas under manipulation the influencing player impacts the other player's *utility* through his actions, under signaling he stands to change her *beliefs*. Consider an example involving,

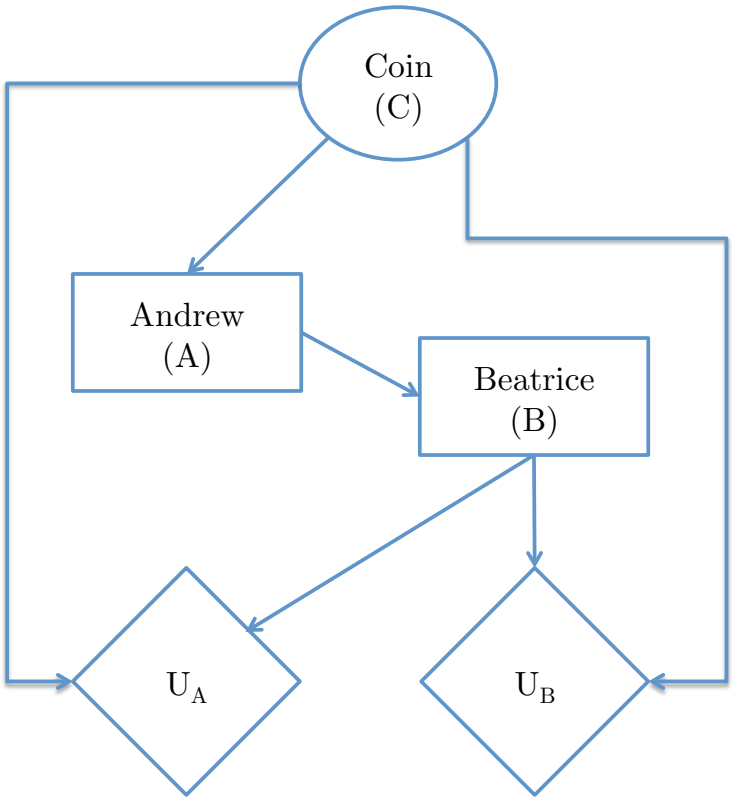

Figure 5.5: An example of signaling

once more, Andrew and Beatrice: A fair coin is tossed and the outcome is revealed secretly only to Andrew, who must then choose between Heads and Tails. Beatrice observes Andrew's choice, but not the outcome of the coin toss, and then also chooses between Heads or Tails. After this, players receive payoffs as follows: Andrew receives \$1 if Beatrice selected Heads, and nothing otherwise. Beatrice receives \$1 or loses \$1 based on whether her choice matched the outcome of the original coin toss, which was unobservable to her. Finally, Andrew loses a dollar if that coin landed on Tails, irrespective of what Beatrice chooses. Graphically, the game is shown in Figure 5.5.

Notice that in this game Andrew's action (A) "signals" something about the unobservable variable C to Beatrice, hence the name for that reasoning pattern. Andrew knows that Beatrice will use the observation of his action to infer something about the coin toss, and he can thus select an appropriate action to influence her decision favorably for himself. This is possible because Beatrice "cares" about that unobservable variable, in the sense that it influences her utility, and because Andrew's utility depends on Beatrice's choice. Formally, five things in the graph define signaling for decision D of agent i: (a) a directed, decision-free path from D to decision node D' of some other agent $j \neq i$; (b) a directed path from D' to a utility node U_i of agent i; (c) an undirected path $< X, \ldots, U_j$ from a parent X of D to some utility node U_j of agent j that is not blocked by $W_{D'}^{D} \cup \{D'\}$; (d) a variable C along that path such that C has diverging arrows, called the "key" node; and (e) a directed path from the key node C to a utility node U_i of agent i that is not blocked by $W_{D}^{C} \cup \{d\}$. The paths in (c) and (e) may contain decision nodes of other agents.

To intuitively understand why these paths are necessary, consider what would happen if any of them were missing. If path (a) were not present, player i could not influence player j at all. Similarly, if (b) were absent, then player i would not have a reason to influence j's choice. If (c) and (d) were absent, then player i would possess no information of value to j, therefore he would have nothing to signal. The variable being signaled is essentially the key node C. Finally, if (e) were missing then player i's action would not depend on the key node C under WD strategies. Since we are operating within the reduced MAID, if (e)

were missing then we would have to sever the path concocting the key node (or its descendant along the path, X) to node D, thus preventing player i from conditioning on it, in which case player j would not make any inferences about the key node from i's observed action.

4. The fourth and final reasoning pattern type is *revealing-denying*. This last reasoning pattern type is similar to signaling, with one exception. Instead of the agent conveying to another something about information he privately—directly or indirectly—observes, he influences another agent's access to information that he does not observe. In other words, the influencing agent controls another's access to information, granting such access (revealing the information to her) or blocking access (denying her the information).

   Again, consider an example. A coin will be tossed (variable C). Andrew gets to choose between Heads and Tails before the coin toss. He must also select whether Beatrice will be allowed to observe the coin toss outcome or not. Beatrice gets to play after to coin is tossed, and must select Heads or Tails. In the end of the game, players get \$1 if their choices matched, and lose \$1 if they did not. Andrew makes a bonus \$10 if Beatrice chose Tails, and Beatrice makes a bonus \$10 if her choice matches the outcome of the coin toss. The MAID for this game is depicted in Figure 5.6.

   The variable C represents the coin toss, and can be either Heads or Tails. Variable E, which represents Beatrice's observation of the coin toss, is distinguished from C, because it can get values Heads, Tails, or Unobservable, in case Andrew denies her access to the outcome of the coin toss. (Andrew's decision now

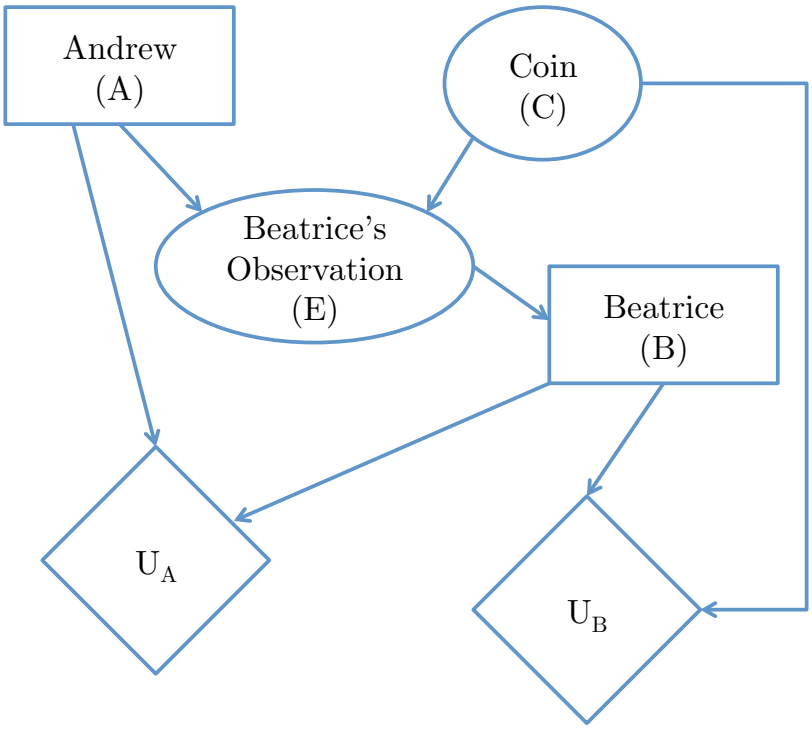

Figure 5.6: An example of revealing-denying

has domain containing four choices, reflecting his dual decision: 'Heads & Reveal,' 'Heads & Not Reveal,' 'Tails & Reveal' and 'Tails & Not Reveal.') Notice how Andrew, with his choice of action, controls Beatrice's access to important information, namely, the coin toss.

Formally, to have revealing-denying for decision D of agent i, three things must be true in the MAID graph: (a) a directed, decision-free path from D to a decision D' of some agent j (here j could be the same agent as i); (b) a directed path from D' to a utility node U_i of agent i; (c) a path from a child of D to a decision node U_j of agent j that is not blocked by $W^D_{D'} \cup \{D'\}$. Paths (b) and

(c) might contain other decision nodes. Intuitively, if (a) or (b) are missing, then as before agent i has no way or no incentive to influence j's decision in D'. As for the third path, its existence signifies that j cares about a variable along that path (it impacts her utility U_j, agent i's action impacts that variable or interferes with her ability to infer it, and she cannot unilaterally infer that variable by means of her own observations (or the conditional blocking set would not be blocking that path).

These four types of reasoning patterns are appealing because they offer a way to intuitively comprehend the objectives of agents. An agent either wishes to improve its own utility directly, or somehow alter the decision of another agent, either by exerting influence on that agent's utility or by controlling its beliefs and observations.

5.2 Extending the Reasoning Patterns to Bayesian Games

In defining the reasoning patterns, Pfeffer & Gal restricted their attention to games of perfect and imperfect information. Although games of incomplete information (Bayesian games) are useful in modeling and studying a large number of interesting strategic situations, reasoning patterns were not defined for them. One notable exception is Bayesian games with a common prior. When the *common prior assumption* (CPA) holds, a game of incomplete information can be transformed into a corresponding game of imperfect information, for which reasoning patterns can be defined.

The CPA has been extensively invoked as a useful and convenient modeling decision. When it holds, the game can be represented in a more concise form, agents' beliefs are mutually consistent, and solving the game is simpler. As noted by a number of economists, however, the CPA is not always appropriate, and might distort the structure of the game being represented. It might also be unrealistic, or simply unnecessary [87].

This section presents an augmented theory of reasoning patterns for all types of games, including games of incomplete information with and without a common prior. Following a discussion on the merits and shortcomings of the CPA, the theory is presented in two steps: First, a new graphical formalism is introduced for both types of Bayesian games, which allows players' beliefs to be represented in a concise way and dependencies between them to be easily detected by constructing the game's *belief graph*. This belief graph is utilized in the second step, in which the reasoning patterns are defined for Bayesian games. Surprisingly, the reasoning patterns defined in this expanded set of games are semantically identical to the four types identified originally by Pfeffer & Gal in games of perfect and imperfect information. However, all four types of reasoning patterns now correspond to significantly richer graphical structures involving the MAID and the belief graph of the game.

5.2.1 The Common Prior Assumption (CPA)

The common prior assumption (CPA) in Bayesian games stipulates that players' types are drawn from a distribution, called the "prior distribution," that is common knowledge to all players. Since the type of a player encapsulates all the information

privately known by her, the CPA essentially implies that players, before learning their own type (i.e., in the absence of any information privately revealed to them), must agree on the likelihood of every stochastic event in the world. More specifically, the typical interpretation of games in which the CPA holds goes as follows: Players originally agree on the game being played and the likelihood of all possible events, which is captured by the common prior. Before any moves are made, players receive observations stochastically by Nature. This information constitutes each player's type, and they use it to condition the common prior to a posterior, which might be different for each agent. After this, players implement their strategies and take actions.

According to Morris [87], the CPA has been widely accepted in the literature on the basis of three arguments:

- *Rationality implies the CPA.* This argument states that rational agents having received identical information should have no reason to disagree on the likelihood of any event or outcome in the world—if one agent disagrees, then she must have made a mistake and therefore she is not thinking rationally. The above argument is flawed in three ways: First, probabilities of events are not directly observable in the world, but are instead statistical properties of complex phenomena that either we do not fully understand or might be too hard to exhaustively model. For example, the fact that a fair coin lands on either side with equal likelihood is a result of small forces applied to its surface, its weight distribution and small air currents in a "typical" environment where the coin is tossed; but the number 0.5 is not directly observable or measurable on the coin

itself. If probabilities are then derived from statistical assessments, rationality objections are irrelevant. Second, if one adopts the subjective view of probability (as Bayesian statisticians do), there is nothing implying that all agents must agree on the probabilities of all events—otherwise probabilities would not be truly subjective. Finally, rationality tells us how to update a prior given new observations, but in general says nothing about how to choose a prior in the first place.

- *Learning will cause agents' priors to converge to the empirically observed frequencies.* This argument basically states that, even if agents start with different and unequal priors, they should, given enough data, update those to more or less identical posteriors, which they will then use as their new (almost common) priors. Of course such a claim is valid only if we can safely assume that agents have been given enough time in the environment to explore and assess the likelihood of every event with sufficient accuracy. In many multi-agent systems, however, the environment is too complex to even represent in state-space, let alone fully explore and quantitatively assess. Furthermore, agents often engage in localized interactions, and therefore their experiences may be concentrated in only a narrow subspace of the environment.

- *Abandoning the CPA would cause serious modeling issues.* This is a more serious concern. As mentioned before, under the common prior assumption the only source of disagreement regarding the likelihood of an event in the world can be private information that some of the agents have obtained but not others. This private information is essentially what the type of an agent represents.

> Therefore, under the CPA agents' inferences are "compatible," e.g., what agent i believes agent j of a particular type believes is exactly what agent j of that type in fact believes. On the contrary, dropping the CPA may lead to infinitely nested beliefs of the form "i believes that j believes that k believes that i believes" that are quite hard to formally reason about computationally [49]. Another problem with such belief structures is that repeated Bayesian games without the CPA may never converge to any equilibrium under any learning process [34].

In many cases, forcing the agents to have a common prior is an unnatural imposition on the model. Why would they have a common prior if they have different subjective beliefs about how the world works, or the space of possible strategies? In such cases, we can try to relax the CPA. To do so, we assign the agents different prior distributions over the joint types. Consider a simple example of a cards game with three players, in which player can be of two types: bluffer, or non-bluffer. The CPA would require that, in the beginning of the game, all players agree on the probability of each player being a bluffer. If we relax the CPA, we will be able to say things like "Alice thinks Bob is a bluffer with probability 0.4, but Carol only thinks he is a bluffer with probability 0.1." The question then could arise: "What does Alice think that Carol thinks about Bob?" One possible answer to this question is to say that while the priors are uncommon, they are common knowledge, so Alice knows Carol's prior and the fact that she thinks Bob is a bluffer with probability 0.1. However, if we are not assuming that the agents have a common prior, this assumption is very hard to justify. If the agents have different subjective beliefs that lead them to have

different priors, how can we expect them to know each others' subjective beliefs?

Alternatively, we can say that agents do not know other agents' priors. Instead, they maintain beliefs about them. One possibility is to say that agents believe other agents' priors are *like their own*, so Alice believes Carol thinks Bob is a bluffer with probability 0.4. In this approach, each agent solves a separate Bayesian game with a common prior equal to their own individual prior. In this way the agents do not reason about other agents having different priors from their own. Each agent comes up with a different, completely subjective solution to the game under this assumption. Obviously, agents whose beliefs about each other's priors are entirely independent might also derive a completely different set of Bayes-Nash equilibria for the game.

Another possibility is to say that each agent has private beliefs about the priors of other agents, which need not necessarily be the same as their own. For example, Alice might believe that Carol thinks Bob is a bluffer with probability 0.3, while Carol herself believes he is a bluffer with probability 0.1. We then need to ask what Alice believes Carol believes Alice thinks about Bob. We thus end up with a complex hierarchy of beliefs that is very hard to model. Furthermore, each agent's belief hierarchy has no interaction with the other agents, so each agent's solution of the game is again completely subjective.

These three approaches—common knowledge, believing everyone has the same prior, and completely private beliefs—are just three out of many structures that can arise when relaxing the CPA. Below, a formalism is presented that can represent a wide and diverse array of belief structures. Before doing that, however, let us look at an example illustrating how such rich structures can come about.

(t_1, t_2)	probability
(maximizer, maximizer)	p^2
(maximizer, naive)	$p(1-p)$
(naive, maximizer)	$p(1-p)$
(naive, naive)	$(1-p)^2$

Table 5.1: Your prior for the RPS game

Suppose you (player 1) are playing Rock-Paper-Scissors (RPS) with a child (player 2). Let us also assume that there are only two types of agents in RPS: 'maximizer' and 'naive.' The maximizer agents have a correct model for the game, and utility function that gives them $+1$ for winning, -1 for losing and 0 for a draw. The naive type, on the other hand, represents how children might be expected to play the game: naive players model their opponent as an automaton that always repeats its last move. Therefore, naives best-respond to their opponent's last choice, e.g., they play 'scissors' after their opponent has played 'paper.' For completeness, assume that naives on the first round choose any action with equal probability.

Now let us assume that you are a maximizer (you know your type, t_1) and you believe that the child's type t_2 is either a maximizer (with probability p), or a naive (with probability $1-p$). This belief is consistent with many possible priors. One possibility is shown in Table 5.1.

However, the child does not share your prior. This is because, if a child is a naive, it would be rather unreasonable to expect it to even be aware of the concept of a game-theoretic maximizer. Hence, the child's prior could look like Table 5.2.

Clearly, no common prior can adequately capture this situation. Many instances in which bounded rationality is evoked, or in which some agents are only partially

(t_1, t_2)	probability
(maximizer, maximizer)	p^2
(maximizer, naive)	0
(naive, maximizer)	$p(1-p)$
(naive, naive)	$1-p$

Table 5.2: The child's prior for the RPS game

aware of the game's structure, can be viewed as generalizations of the above scheme. The question that now arises is "what prior does the child believe you are using?" One option here would be to assume that the child believes that you use your actual prior, the one shown in Table 5.2. In this case, the situation is a game with uncommon, common knowledge priors. However, closer scrutiny would reveal that this turns out to be rather unnatural: the child, when its type is naive, is assumed not to even be aware of the possibility of you being anything but naive, but in this model its also maintains a non-zero probability that you might think it is not a naive.

A better solution here can be achieved by constructing a richer belief structure. In this case, we shall use three types: 'maximizer,' 'naive,' and 'naive-adult.' An agent of the naive-adult type models the game exactly as a naive, but disagrees with the naive type on the prior. In our example, you, as an adult, will still use the prior shown in Table 5.2. The child, however, will use the prior of Table 5.3 (missing assignments are assumed to have probability zero).

Furthermore, the child shall believe that you are using the prior in Table 5.4. This belief structure now describes a child that, if a maximizer, will be aware of the possibility that you are either a maximizer or a naive (Table 5.3, first and third rows).

(t_1, t_2)	probability
(maximizer, maximizer)	$0.9p^2$
(maximizer, naive)	0
(naive, maximizer)	$0.9p(1-p)$
(naive, naive)	0
(naive-adult, naive)	$0.9(1-p)$
(naive-adult, naive-adult)	0.1

Table 5.3: The child's updated prior for the RPS game

(t_1, t_2)	probability
(maximizer, maximizer)	$0.7p^2$
(maximizer, naive)	$0.7p(1-p)$
(naive, maximizer)	$0.7p(1-p)$
(naive, naive)	$0.7(1-p)^2$
(naive-adult, naive-adult)	0.3

Table 5.4: The child's updated prior for your prior in the RPS game

If the child is a naive, though, the child will also assume that you are a naive (Table 5.4, fifth row) and that you are as oblivious to the existence of any types other than naive. To capture that last piece, the naive-adult type is introduced, such that it is wholly similar to the naive type, except it fails to consider other alternative types (Table 5.4, last row).[4]

5.2.2 Graphically representing Bayesian games

The main idea behind the formalism is that modeling a game becomes simpler if the players' types and beliefs are captured in a conceptually appealing and graphical way. As shown later, this allows the agents' beliefs, even when the CPA does not

[4]Note: The constants in Table 5.2 – 5.4 are chosen such that the probabilities sum to one and do not affect the posterior distributions.

hold, to be represented concisely. It also allows agents to reason about which beliefs of other agents they need to consider in their own decision-making problem, instead of having to reason about the (possibly vast in number) sets of beliefs of all other agents. Finally, it allows reasoning patterns to be defined for Bayesian games without a common prior, which can be used to answer qualitative questions about the behavior of agents, as well as identify, anticipate or prescribe well-performing strategies, as discussed in Section 5.2.3.

The formalism introduces the concept of a "block," and defines a game as a collection of blocks B. A block $b \in B$ consists of two elements: (1) the model $m(b)$ players in that block have about the world, and (2) the beliefs $\beta(b)$ the players assume, and believe others to have, in that block. The players' model is a complete game in normal or extensive form, with every player's information sets, available moves and utilities fully specified. The beliefs in block b consist of $n(n-1)$ probability distributions over B, indexed p^b_{ij} for all $i, j \in N, i \neq j$, where N is the set of agents ($|N| = n$). The distribution p^b_{ij} captures agent i's beliefs over which block agent j is using. Also, let us denote by $p^b_{ij}(b')$ the probability assigned to block b' by the distribution p^b_{ij}.

It is straightforward to map this construct onto a Bayesian game. For each agent i, her typeset T_i is equivalent to the set of blocks B. When agent i is of a particular type, say $b \in B$, then agent i's private information (utility, observations, etc.) are fully captured by the game $m(b)$. Moreover, i's posterior distribution over the beliefs

of all other agents given her type, $p(\mathbf{T}_{-i} = (t_j)_{j \neq i} | T_i = b)$, is given by the product:[5]

$$\prod_{j \neq i} p_{ij}^b(t_j)$$

In each block b, the set of pure strategies for player i contains all her pure strategies in the model $m(b)$. For the game as a whole, a pure strategy for i is then a choice of pure strategy for every block $b \in B$. In other words, i's pure strategy space is the product space of her pure strategies across all blocks $b \in B$.

Furthermore, if the models $m(b)$ are represented in extensive (tree) form, a pure strategy for i for the whole game is a mapping from all information sets of all trees $m(b)$ to an action available to her in every such information set. Similarly, mixed strategies are probability distributions over pure strategies, and behavioral strategies can be defined as mappings from information sets to probability distributions over available actions. Finally, as in typical Bayesian games, a strategy profile σ denotes, for every agent i and every type $b \in B$, a choice of mixed (or behavioral) strategy $\sigma_{i,b}$.

The belief graph

One useful property of the formalism is that it allows for belief dependencies to be uncovered easily. In particular, it can help a modeler answer the question "Which of the beliefs of other agents are relevant to agent i's decision-making?" Alternatively, this question can be stated as "Which of the other agents' beliefs does i need to know

[5]Notice how in the formalism the modeling is performed in terms of the posterior distributions $p_i(T_j | T_i = b) = p_{ij}^b$, not the priors $p_i(T)$. Given these posteriors, any prior that is consistent with them will be essentially expressing the same game.

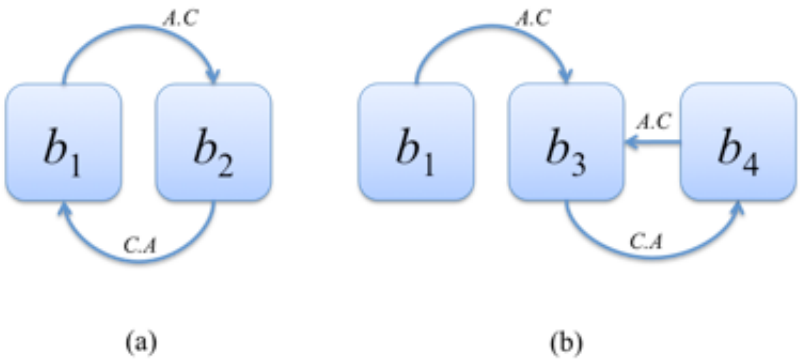

Figure 5.7: The belief graph of the RPS game

in order to compute her optimal decision?" This is performed by constructing the game's belief graph.

In Figure 5.7 the belief graph for the Rock-Paper-Scissors example discussed above is presented. Blocks are represented as nodes and edges (x, y) are labeled "$i.j$" to indicate that agent i in block x believes agent j to be using block y (i.e., agent j is believed to follow the structure of the game and adopt the beliefs within block y). Figure 5.7(a) shows the graph for the case of common knowledge priors. The block b_1 contains game $m(b_1)$, which is a standard description of rock-paper-scissors, while the block b_2 contains game $m(b_2)$, an alternative game in which the adult always repeats her last move. In b_1, you (the adult) model the child as using b_2, while in b_2 the child models you as using b_1. Next to this graph, in Figure 5.7(b), the case with the richer belief structure is illustrated, whereby you model the child as using b_3, and the child models you as using b_4.

The belief graph reveals which belief sets of others an agent needs to take into

account in its decision-making. Notice how in the case with individual priors (Fig. 5.7(b)) the child, which follows b_3 need not be aware of your beliefs in b_1 to compute its optimal strategy, since it considers you to be using block b_4. On the other hand, you in b_1 must consider the child's beliefs in b_3, and therefore also your own beliefs in b_4. These insights about belief dependence can be obtained by reachability arguments in the graph. The only blocks (games and beliefs) relevant to you are the ones that are reachable from your own block, following directed edges.

Formally, the belief graph is constructed as follows: Its nodes are the set of blocks B. Then, we add an edge (b, b') and we label it "$i.j$" if $p^b_{ij} > 0$. This edge denotes that agent i in block b assumes that j might be using block b' as his model of the world. The destination block b' may be the same as the source b (self-edge). Next, we define a path $\pi = (b_1, \ldots, b_m)$ such that, for every node b_k, where $k \in [1, m-1]$, there is an edge (b_k, b_{k+1}) and, for each consecutive edge pair $\{(b_k, b_{k+1}), (b_{k+1}, b_{k+2})\}$, where $k \in [1, m-2]$, the label of the first edge is "$i.j$" and the label of the second is "$j.k$" for some agents i, j and k. (A path may very well contain self-edges.) We say that a block-agent pair (b', j) is reachable from pair (b, i) if there is a path from b to b' in which the first agent is i and the last agent is j. The set of reachable blocks from (b, i) is denoted by $R(b, i)$.

Only those posterior distributions $p^{b'''}_l$, where $(b''', l) \in R(b, i)$, are relevant to agent i's decision-making, when that agent is in block b. This is because, in block b, when this is deemed by agent i to be the "true world" (that is, b is the true type of agent i), some other agent j will be modeled as if he was using one of the blocks for which an edge (b, b') exists with label "$i.j$," hence j's beliefs in b' need to be considered by

i in b. Furthermore, agent j in b' might be modeling k (who could be the same as i), as if she were using some block b'', which j needs to consider in order to predict their behavior and hence best-respond to it. Therefore i in b, who is best-responding to j, must also consider k in b''. By induction, if $(b''', l) \in R(b, i)$, the game tree $m(b''')$ and the posterior $p_l^{b'''}$ are potentially relevant to i's decision-making problem. On the other hand, if $(b''', l) \notin R(b, i)$, there is no path of reasoning by which agent i in b needs to take agent l in b''' into consideration, so $p_l^{b'''}$ is irrelevant to i in b.

In addition, if there is an edge from b to b' labeled "$i.j$," then agent i in block b believes that agent j's model of the world is $m(b')$ and his beliefs are $\beta(b')$. Thus agent i in b knows the model and beliefs of j in b'. Likewise, if there is an edge from b' to b'' labeled "$j.k$," agent j in b' knows agent k's model and beliefs in b''. It follows that agent i in b knows agent k's model and beliefs in b''. Recursively, agent i in block b knows the beliefs of all agents l in blocks b''' such that $(b''', l) \in R(b, i)$. Thus the belief graph precisely captures what agents must know about the beliefs of other agents. A subtle point must be made, however. If (b', j) is not reachable from (b, i), the graph does not preclude the possibility that agent i in b knows the beliefs of agent j in b'. It merely says that i does not *need* to know them in the context of estimating her optimal decision.

Reachability in the belief graph is equivalent in practice to *knowledge*. Since unreachable parts are irrelevant to an agent's decision-making problem, they can be considered unknown. Inversely, if an agent is not aware of another agent's beliefs, they will be unreachable from his block in the belief graph.

This allows us to represent the different relaxations of the CPA discussed above

as special cases of the belief graph. If in the graph every block-agent pair is reachable from every other block-agent pair, then the agents' beliefs are common knowledge. This is because, when every block is reachable from every other block, there are no agents (no matter which block they exist in) that have partial knowledge or partial visibility of another's beliefs, hence everyone's beliefs are mutually consistent. The case in which priors are completely private is captured by a belief graph containing n disconnected subgraphs, one for each agent. In this extreme case, not only are the agents' beliefs divergent, but they have no information whatsoever regarding others' beliefs. All other graphs represent in-between cases, in which agents might have individual beliefs, but also some knowledge about others' beliefs that are relevant to them.

An extended example

An extended example is now presented that further illustrates these concepts, and shows the interesting equilibrium analysis that the framework supports. Suppose Alice works for company X and all her retirement savings are tied to X's stock, therefore she would benefit greatly from an increase in their value. Bob is a billionaire considering to buy a large number of X's stocks, but lacks the expertise to make this decision. Hence, he relies on a predicting agency C that informs him whether the stock price will go up (in which case it is optimal for him to buy) or down (in which case he should optimally rest). Clearly, Alice would significantly prefer Bob buying the stock. Let us also assume that Bob believes that C, the predicting agency, has prior probability 0.2 to suggest 'buy' and 0.8 to suggest 'rest.'

Suppose now that, in Alice's model of the world, she can "threaten" the predicting agency in some frowned-upon way, which Alice thinks is entirely effective, i.e., a threatened C will do as Alice says, i.e., suggest 'buy' if Alice dictates it to say 'buy,' and suggest 'rest' if Alice dictates it to say 'rest.' (Assume that either action has zero cost for Alice.) In Bob's model, Alice can choose either action, but the agency cannot be bullied by her threats (in this case, the scenario is equivalent to Bob being un-aware of Alice's presence); this means that Bob considers, in his model, C's suggestion to be trustworthy and predictive of the stock's future price. Alice, however, is uncertain of whether Bob is aware of her manipulative powerin fact, she believes that the probability of him being aware of her actions' effects is 0.3.

We shall create two blocks to represent this situation ($B = \{K, L\}$). In blocks K and L the models, which are represented by extensive form game trees, look like in Figure 5.8. At first Alice decides whether to dictate 'buy' or 'rest.' Then Nature, labeled C, representing the agency, makes a suggestion to Bob. Finally, Bob has to decide whether to buy the stock or not. He has two information sets, one representing the history "C suggested 'buy'," and one capturing the alternative "C suggested 'rest'." The only difference between the two trees $m(K)$ and $m(L)$ is the fact that the probability of C making a 'buy' suggestion to Bob in $m(L)$ is 0.2 independently of Alice's action, whereas in $m(K)$ it is either one or zero, depending on Alice's threat.

Let us now define the beliefs β. In block K we need to have a distribution $p^K_{A,B} =< 0.3 : K, 0.7 : L >$ to capture the fact that Alice of type K thinks that Bob is likely to use K (and hence be aware of her threats) with probability 0.3, or use L (and therefore be oblivious to her actions) with probability 0.7. Bob in K will have

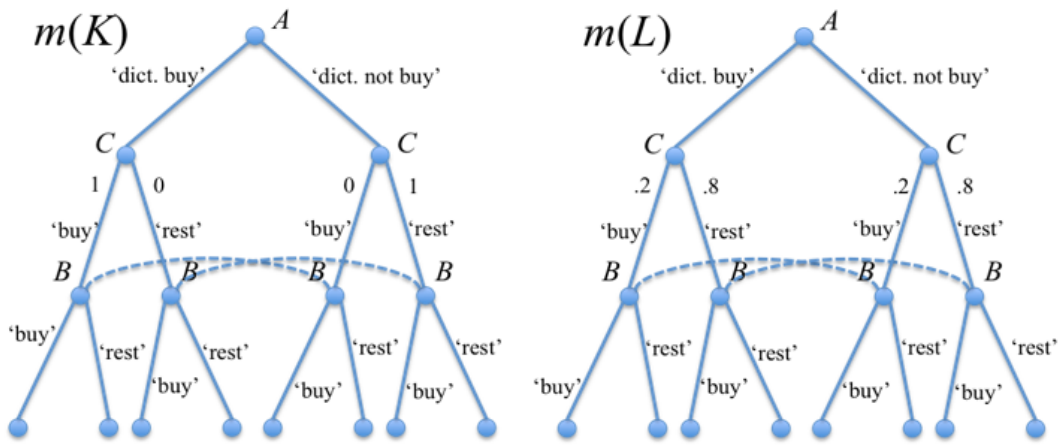

Figure 5.8: The trees for the Alice-Bob example

what we call a trivial belief, i.e., $p^K_{B,A} =< 1 : K >$, meaning that a Bob who is aware of Alice's threats also considers her to be aware of them, which makes sense. In block L both agents have trivial beliefs, such that Bob of type L is unaware of Alice.

The belief graph for the game is presented in Figure 5.9. Notice here how Bob of type L has no edges going back to K, hence Bob of that type need not even consider (or be aware of what happens in) block K. This makes sense, as Bob in L is modeled as an agent who is truly oblivious of Alice's threats.

Identifying the game's Bayes-Nash equilibrium could be done as follows: First of all, Alice has two pure strategies in either block, 'dictate buy' and 'dictate rest.' Bob in turn has four strategies, 'do what C suggests,' 'do the opposite of what C suggests,' 'always buy' and 'always rest.' In tree $m(L)$ it is clearly optimal for Bob to 'do what C suggests,' as the agency's suggestion is predictive of the stock's performance. And of course, Alice cannot affect the situation, so any randomization between 'dictate

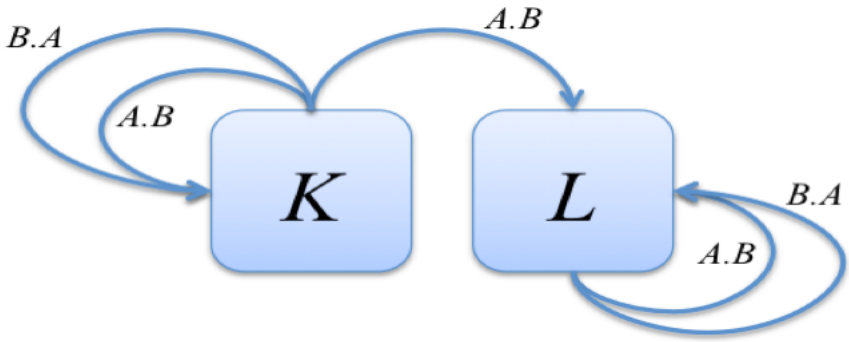

Figure 5.9: The belief graph for the Alice-Bob example

buy' and 'dictate rest' constitutes an equilibrium strategy. In block K, however, things are different. First, the strategy 'do the opposite of what C suggests' is weakly dominated by 'never buy' for Bob; once re-moved, then 'dictate buy' becomes weakly dominant for Alice, in which case the dominant strategy for Bob becomes 'never buy.'

These would be the equilibria in each block, if that block was commonly held to be true by both agentsbut how about the game as a whole, given the agents' complex beliefs? Clearly, the Bayes-Nash equilibrium would have Bob of type L playing 'do what C says,' as in that case Bob is unaware that Alice might even be of a different type. For Bob of type K, similarly, he should 'never buy,' as his beliefs in K are also trivial. As for Alice, if she is of type L she could do whatever, but if she is of type K, she needs to best-respond to a mixture consisting of Bob of type K (with 0.3 probability) and Bob of type L (with 0.7 probability). Hence, her optimal strategy overall would be to 'dictate buy.'

Notice that the equilibria are interdependent but subjective. We have not taken a position which of these blocks, K or L, accurately represents the real world (if any

does). The notion of "truth" is not employed in the analysis. If K happens to be true, then Alice's threats will indeed be effective and Bob of type L will be most likely misled into an ill-adivsed purchase. If L happens to be true, there will be no effect and Bob's decision will be profitable to him. If it was important to us to identify what would happen in the real world, we could introduce a modeler agent whose beliefs correspond to what we believe the truth actually is.

5.2.3 An augmented theory of reasoning patterns

If the CPA holds in a Bayesian game, the game can be represented in the graphical formalism as a single block. Within this block, the joint type $\mathbf{T}$ can be a variable in the MAID representation of the game, and individual types T_i can be deterministic children of $\mathbf{T}$. Then, we can draw an edge (T_i, d) for all decisions d of agent i, and the representation is complete, rendering the Bayesian game as a game of imperfect information. As far as the reasoning patterns are concerned for such games, since Bayesian games with a common prior are then representable in a single MAID, the original definition of the reasoning patterns can be used unchanged.

However, as soon at the CPA is relaxed, the original definitions of the reasoning patterns cannot be applied. As we have seen, in a Multi-Agent Influence Diagram (MAID), the directed edges represent probabilistic dependencies between variables (actions, events, utilities). In the block formalism, we say that agent i "believes" in edge (x, y) of a MAID if this edge exists in the model $m(b)$ of the block b she has been assigned to (i.e., the block that represents his real type). Clearly, if all the paths constituting a reasoning pattern exist wholly within the block of agent i, then that

reasoning pattern is believed by i. However, a reasoning pattern may span several blocks.

Before we proceed, let us remember the definition of a *motivated* decision and that of *well-distinguishing* (WD) strategies. A decision D of agent i in block b is motivated if, when i has been assigned to block b, there is a strategy profile for all other agents σ_{-i} such that there are two actions a_1 and a_2 available in D that yield a different expected utility: $\mathrm{E}[u_i^b(a_1|\sigma_{-i})] \neq \mathrm{E}[u_i^b(a_2|\sigma_{-i})]$. In simpler terms, a decision is motivated for an agent if her action "matters" under some choice of strategy by other agents. (On the contrary, if a decision is not motivated, then any choice by i in that decision is equally good, so his choice "does not matter.")

A strategy σ_i for player i defines, for each decision D of i, a mapping from value assignments to its parents $\mathbf{Pa}(D)$ to a probability distribution over actions a_j available in D. A strategy σ_i is then well-distinguishing (WD) for i if, intuitively, i does not condition on a variable $v \in \mathbf{Pa}(D)$ if that variable has no effect on her utility (hence, "well-distinguishing"). Hence, if configurations c_1 and c_2 of $\mathbf{Pa}(D)$ differ only in the value of variable V, a strategy would be well-distinguishing if it prevented i from choosing a different probability for any action a_j under c_1 and c_2, unless her expected utility was different, i.e., V "mattered."

The four reasoning patterns are then defined, examples of which are given in Figure 5.10. In the definitions below the following assumption is made: For every node c in block b we assume there is a corresponding node c in every other block b', i.e., MAIDs in all blocks have the same set of nodes. Only edges are allowed to differ between blocks. In the following, we denote as $\mathbf{U}_i$ the set of i's utility nodes.

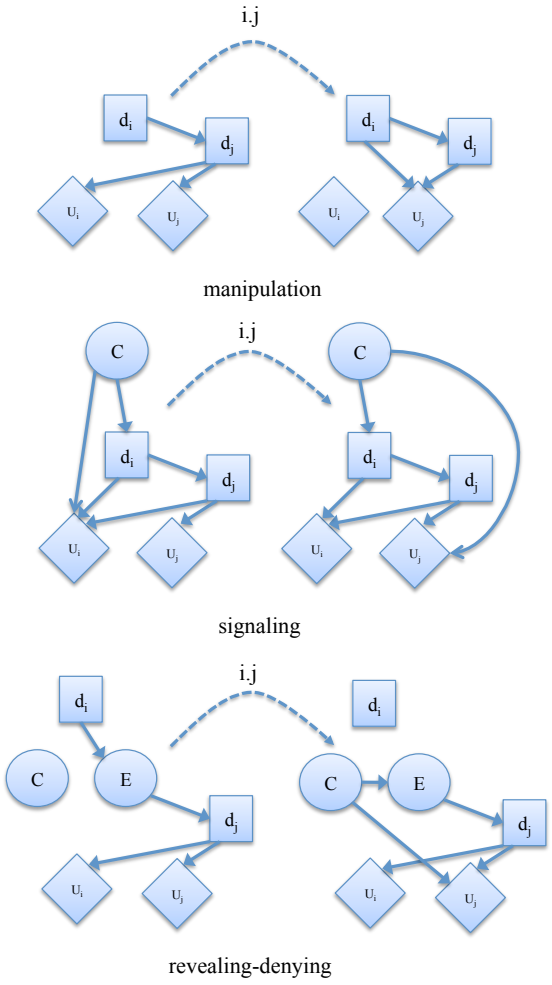

Figure 5.10: Examples of reasoning patterns across blocks

1. A decision d_i of agent i in block b has **direct effect** if there is a directed path wholly within $m(b)$ from d_i to one of the utility nodes of i that does not go through any other decision node (decision-free path). Direct effect captures the influence an agent's action has on her utility, without the intervention of other agents.

2. A decision d_i of agent i in block b has **manipulation** if there exists:

   (a) a directed, decision-free path in b from d_i to a decision d' of agent $j \neq i$,

   (b) a directed path from d' to a node in $\mathbf{U}_i$ in b that contains only motivated decision nodes (effective path), and

   (c) a directed effective path from d_i to a node in $\mathbf{U}_j$ (the utility nodes of j) that does not go through d' in some block b' for which $p^b_{ij}(b') > 0$, i.e., which is considered possible for j by i in b.

   Manipulation captures how an agent can influence her utility through exercising influence on another person's utility. Agent i takes an action that influences j's utility (path c). This changes j's optimization problem, because j stochastically knows i's action (path a). His (j's) optimal action under this setting can thus be influenced to increase i's utility (path b).

3. A decision d_i of agent i in block b has **signaling** if there exists:

   (a) a directed, decision-free path in b from d_i to a decision d' of agent $j \neq i$,

   (b) a directed effective[6] path from d' to a node in $\mathbf{U}_i$ in b,

[6]As explained above, a path is called "effective" if all the decision nodes on it are motivated.

(c) an undirected effective path from a node C to a utility in $\mathbf{U}_j$ that is not blocked by the set $W_{d'}^{d} = \{d'\} \cup (\mathbf{Pa}(d') - \mathbf{Desc}(d))$, where $\mathbf{Desc}(d)$ are the descendants of node d, in some block b' for which $p_{ij}^{b}(b') > 0$,

(d) a directed effective path from C to d_i in both b and b', and

(e) an undirected effective path from C to a node in $\mathbf{U}_i$ in b, that is not blocked by W_{d}^{C}.

Signaling captures the situation in which agent i can influence her utility through the action of another agent j, by conveying something about a variable in the world (C) that she can infer (path d) and which j cares about (path c).

4. A decision d_i of agent i in block b has **revealing-denying** if there exists:

(a) a directed, decision-free path in b from d_i to a decision d' of agent $j \neq i$,

(b) a directed effective path from d' to a node in $\mathbf{U}_i$ in b,

(c) a directed effective path from a node E to d' in in some block b' for which $p_{ij}^{b}(b') > 0$,

(d) a directed effective path from d_i to E in b,

(e) a directed effective path from a node C to E in b and b', and

(f) a directed effective path from C to a node in $\mathbf{U}_j$ in b' that is not blocked by $W_{d'}^{d}$.

Revealing-denying captures the situation in which an agent (i) influences her utility thought the action of another agent (j) by controlling the uncertainty j has over some variable (C) that he cares about. By increasing or decreasing

the clarity by which j can infer C through E (paths d, e), agent j's optimal decision can be changed to benefit i's utility (path b).

We say that a decision node that has a reasoning pattern is "effective." The following theorem is then proved:

Theorem: *If a decision node d of agent i in block b is motivated, and all agents use WD strategies, then it is effective.*

PROOF: First, we look at our restriction to WD strategies, which implies the following for our graph structure. First, if a parent node v of a decision d has no effect on i's utility (for any assignment to the MAID paremeters), then is it d-separated from $\mathbf{U}_i$ given the set $W_d^v = \{d\} \cup (\mathbf{Pa}(d) - \{v\})$. Then, if i uses only WD strategies, she does not condition on such a variable v, therefore we can sever the edge (v, d) and retain an equivalent MAID graph.

Suppose then for the sake of contradiction that d is motivated but not effective. Since d is not effective, there is no directed, decision-free path from d to $\mathbf{U}_i$ in b (by definition of direct effect). Therefore, the only way d might affect $\mathbf{U}_i$ in b (in order to be motivated) is through some other agent $j \neq i$. Suppose d' is the decision node of agent j that facilitates this indirect effect. By our assumption, d must be able to affect d', therefore there must be a directed path from d to d' in block b; moreover, d' must be able to affect a node in $\mathbf{U}_i$, hence there must be a directed path from d' to $\mathbf{U}_i$ in block b.

But since j uses WD strategies, the path from d to d' can only exist if u, the parent of d' along that path, is not severed from d', that is, if u is not d-separated by $\mathbf{U}_j$ given $W_{d'}^u$. The only way this can be is if u is *d-connected* to $\mathbf{U}_j$ given this

set, i.e., there is an undirected path from u to $\mathbf{U}_j$ that is not blocked by $W^u_{d'}$. This path should exist in the MAID of a block which is deemed likely for j to use, i.e., in a block satisfying $p^b_{ij}(b') > 0$.

Such a path might go through d or not. If it goes through d, there can be two cases: it either contains a sequence of nodes $< x, d, y >$ where y is a child of d, or y is a parent of d (by definition x must be a child of d). If y is a child of d, then the definition of manipulation holds, as there is now a subpath from d to $\mathbf{U}_j$ that does not go through d' by its blocking restrictions. If y is a parent of d, then signaling holds. This is because there is now a path from C, an ancestor of d (and y), to a node in $\mathbf{U}_j$. Moreover, since the edge from C to d is retained and not severed, there must be a path from C to some utility $\mathbf{U}_i$, which completes the definition for signaling. The only remaining case is if the path does not go through d at all. Then, there must be an edge E along that path that is a descendant of d, and a sequence of nodes along the path $< x, E, y >$, where y is a parent of E but not on the path from d to E. In that case, there must be a path from y to $\mathbf{U}_j$ that satisfies the blocking properties. But this is the definition for revealing-denying. As a final note, if any decision nodes exist along any of these undirected paths, the edges incoming to them must not be severed, therefore similar restrictions exist, i.e., the nodes must be effective. This is the reason why the definitions of the reasoning patterns use "effective paths." In conclusion, it cannot hold that a node is motivated but not effective, Q.E.D. □

The theorem is significant because it renders the four reasoning pattern types "complete." If a decision of an agent has no reasoning pattern, the theorem states that the decision will not be motivated, and thus the agent will have no reason to

prefer one decision over another. Therefore, by examining the reasoning patterns we can capture all the reasons why an agent might choose an action.

Using reasoning patterns for Bayesian games

The usefulness of reasoning patterns is illustrated in the analysis of Bayesian games by means of an example. Imagine there is an intelligence agency consisting of N agents. These agents collect information in the world, then summarize and interpret it, passing it on to their superiors, who then aggregate all the information and make decisions. Such a domain can be represented by a MAID. Rectangles are the actions taken by the agents, and oval nodes are information collected in the world or passed between agents. If all agents are cooperative, then all can be assumed to share a utility function U, which is represented as a single diamond node in the graph.

However, some of the agents might be "confederates." Such agents are trying to subvert the operation of the agency, and therefore can be assumed to have a different utility function U', which gives them a high value when the agency fails (i.e., when U is low). The agency is aware of the possibility of confederates among its members.

To take a simple case, suppose $N = 4$, named 1, 2, 3 and 4 respectively. Each agent i might be a confederate ($c(i) = 1$) or not ($c(i) = 0$). If agent i is a confederate, we also assume he knows all other agents that are confederates. Finally, we make the assumption that there are either zero or exactly two confederates in the agency.

In a Bayesian game, each agent would have a type t_i, drawn from a set T_i. Each type would have to indicate (a) whether the agent is a confederate, and (b) if the agent is indeed a confederate, the identity of the other confederate. Hence $T_i =$

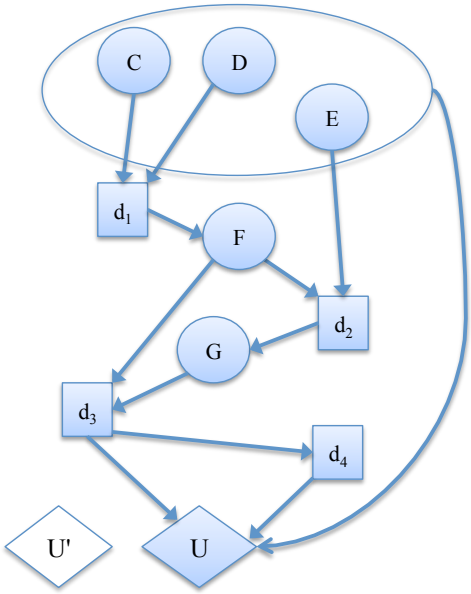

Figure 5.11: Agency example

$\{(c, j) : c \in \{0, 1\}, j \in N \cup \{\emptyset\} - \{i\}\}$, with the restriction that, if $c = 0$ then $j = \emptyset$, and if $c = 1$ then $j \neq \emptyset$. The joint type vector $\mathbf{T} = \times_i T_i$ might be drawn from a common distribution $p(\mathbf{T})$, or not. The latter case, in which the common prior assumption does not hold, might be necessary to describe cases in which some agents trust the various members of the agency more than others, e.g., if the prior of 1 for the possibility of 3 being a confederate is different than the prior held by 2 for the same event.

Imagine now that the graph looks as in Figure 5.11. Agents 1 and 2 get information from the world (C, D and E) and compile reports (F and G). Agent 3 then makes a decision which is being communicated to agent 4, who makes a final decision. The utility node U is influenced by the decisions of agents 3 and 4, but is being shared by

all agents. Agents in this game have reasoning patterns: Agent 4 has direct effect. Agent 3 has both direct effect (edge (d_3, U)) and manipulation (through 4). Agents 1 and 2 have signaling (as they signal the values of C, D and E) to agent 3.

Suppose now we were interested in answering the following question, set forth by agent 4, who is not a confederate: "Which pairs of agents should be more feared to be confederates?" and "Which pairs of agents are more likely to be the confederates, given that misreported information have been observed in node G?" In a traditional analysis, we would have to know, given $t_4 = (0, \emptyset)$, what the distribution of (t_1, t_2, t_3) is and, given this distribution, what the Bayes-Nash equilibria of the game are. Then, we would answer the first question by trying to compare the expected behavior of the players under the various Bayes-Nash equilibria with the observed behavior, as indicated by the misinformation received by player 4.

The problems with this analysis are that (a) there might be a multitude (or infinity) of equilibria, making the comparison hard, (b) equilibria are not easy to compute to begin with,[7] and (c) players might not agree on which equilibrium is to be played, or they might not be rational equilibrium players at all. Furthermore, this analysis requires that we know the probability distributions of all variables C, D, E, F and G, as well as the exact formula in U.

On the contrary, reasoning patterns allow us to do the following: First, we can represent this game in blocks $B = \{b_1 = (0, \emptyset), b_2 = (1, 2), b_3 = (1, 3), \ldots\}$. Each player i is assigned to one of the blocks $B - \{(1, i)\}$, because there must be exactly two confederates if he happens to be one himself. Edges between the blocks are

[7] In this graph computation would be trivial, but in larger graphs it would be significantly more challenging.

drawn accordingly. Next, we can run the (polynomial) algorithm for the detection of reasoning patterns, described in the Section 5.4 of this thesis.

We may then claim that the agents that have reasoning patterns such as manipulation, signaling and revealing-denying are more susceptible to being confederates than other agents. This requires some explaining. In signaling, the value being signaled must not be observable by other agents (otherwise the blocking conditions for path c in the definition do not hold). Hence, if an agent has a signaling reasoning pattern, that means he has the ability to misrepresent information without that being directly detectable. Similarly, through a revealing-denying reasoning pattern he controls access to information other agents have. Agents with this pattern can greatly enhance or impede the decision-making ability of their superiors. Likewise, manipulation reasoning patterns involve fabricating information that is input to some other agent's problem.

But the reasoning patterns do not just tell us that there might be an effect. They tell us "what the effect *is*," e.g., which variable is being signaled, or which variable will contain fabricated information. For instance, in the manipulation reasoning pattern, the confederate (i) will alter the value of the nodes on the path from d_i to d_j, where j is one of his superiors and no intermediates exist between i and j. Hence, if we receive evidence that one of these reports are fabricated, we can immediately cast suspicion upon agent i.

Also notice that the reasoning patterns analysis does not require knowledge of the exact utility function, or all the probabilistic dependencies. But if such knowledge is available, we may quantify the reasoning patterns, and calculate the expected utility

of misrepresenting a variable by a confederate. Still, reasoning patterns would enable us to limit this search within the variables that the alleged confederate would have a reason to maliciously influence through his reasoning patterns.

In general, examining the reasoning patterns is suitable for answering questions of a qualitative nature. We may gain insight on the strategies agents are likely to follow, or understand their reasoning, even without solving the game. We may thus be able to anticipate their behavior, respond appropriately to it, or come up with well-performing strategies. On the other hand, the reasoning patterns do not necessarily offer a way to address more quantitative problems, such as identifying a Nash equilibrium of the game. In complex games, however, it might be very challenging to answer such quantitative questions, especially with limited computational resources or under time pressure. In those settings, the reasoning patterns can offer a shortcut to quickly break down a game and understand or predict the behavior of agents, by precisely avoiding expensive computations.

5.3 Simplifying Games With Reasoning Patterns

Pfeffer & Gal [103] prove a theorem according to which every decision in a MAID for which a player has a reason to deliberate, in the sense that some actions yield her higher utility than others, is associated with at least one reasoning pattern. As a corollary, in every decision that is not associated with a reasoning pattern, the agent is indifferent between all actions. This observation can be used in the context of equilibrium computation. In particular, a game G can be simplified by removing all these decision nodes, before a Nash equilibrium is computed for it. The equilibrium

of the this reduced game, G', can then be extended into a Nash equilibrium for the original game, by merely plugging uniform random strategies for the decision nodes that were removed. In certain classes of games, this simplification can lead to exponential time savings in equilibrium computation.

A useful analogy from game theory is the elimination of dominated strategies in the Normal form representation of a game. A strategy of an agent is strongly (respectively, weakly) dominated when, under any choice of strategy by other agents, it yields strictly less utility (respectively, at most as much as) another strategy of that agent. Clearly, dominated strategies can never be part of a Nash equilibrium of the game, hence they can be eliminated, and doing so may reduce the running time of an equilibrium detection algorithm. Similarly, by removing decision nodes that have no reasoning patterns, we can solve for an equilibrium of a smaller game faster. In fact, removing decision nodes with no reasoning patterns is equivalent to eliminating weakly dominated strategies that differ only in what the agent does in those decision nodes. Using the reasoning patterns, however, this can be done even without enumerating all the strategies of the agent and comparing their payoffs.

An algorithm is presented here (Algorithm 1) that takes a game of perfect or imperfect information, G, in MAID format, and returns a simplified version of it, by removing all edges that do not have a reasoning pattern. The algorithm uses procedures **df**, **man**, **sig**, **rev** and **retract_edges**, which are defined below. The first four of these correspond to detecting the existence of the respective four reasoning patterns, whereas the fifth prunes edges connecting decision nodes to any parents of them which are irrelevant. The algorithm proceeds iteratively, and terminates

Algorithm 1 The simplification algorithm

Input: MAID G

$D \leftarrow$ decision nodes in G
for d in D **do**
  $effective(d) \leftarrow true$
repeat
  $retracted \leftarrow false$; $simplified \leftarrow false$
  {identification phase}
  repeat
    $changed \leftarrow false$
    for d in D **do**
      if not (**df**(d) or **man**(d) or **sig**(d) or **rev**(d)) **then**
        $effective(d) \leftarrow false$
        $simplified \leftarrow true$
        $changed \leftarrow true$
        remove edges incoming to d
        make d a chance node w/uniform distr.
    discard memoization database
  until $changed = false$
  {pruning phase}
  if retract_edges(G) **then**
    $retracted \leftarrow true$
until $retracted = simplified = false$

when a full iteration (identification and pruning phases) causes the graph to remain unchanged. The output is the reduced MAID corresponding to the game.

All these procedures are built upon simple graph reachability or path blocking operations, all of which can be efficiently computed in polynomial time. For example, the procedure **directedDecisionFreePath**(x, y) is simply implemented by a breadth- or depth-first search. Certain more sophisticated procedures are used to search for a path that satisfies particular properties. For example, **effectivePath**(x, y, W) is used to search for a path from x to y on which all decision nodes d have a reasoning pattern (called "effective" nodes) and, moreover, the path is not blocked by the set of nodes W.

df(d)

1: $U \leftarrow$ utility nodes belonging to the owner of d
2: **for** u in U **do**
3: **if directedDecisionFreePath**(u, D) **then**
4: **return** true
5: **return** false

man(d)

1: $U \leftarrow$ utility nodes belonging to the owner of d
2: $N \leftarrow$ decision nodes reachable by d through a directed decision-free path
3: **for** u in U and n in N **do**
4: $U' \leftarrow$ utilities belonging to the owner of n
5: **for** u' in U' **do**
6: **if directedEffectivePath**(n, u) and **directedEffectivePathNotThrough**(d, u', n)

```
        then
7:          return true
8:  return false
```

sig(d)

```
1:  U ← utility nodes belonging to the owner of d
2:  N ← decision nodes reachable by d through a directed decision-free path
3:  for u in U and n in N do
4:    U' ← utilities belonging to the owner of n
5:    w' ← all parents of n that are not descendants of d
6:    for u' in U' do
7:      if directedEffectivePath(n, u) then
8:        A ← ancestors of d
9:        for a in A do
10:         if backDoorPath(a, u', w') then
11:           w ← all parents of d that are not descendants of a
12:           if effectivePath(a, u, w) then
13:             return true
14: return false
```

rev(d)

```
1:  U ← utility nodes belonging to the owner d
2:  N ← decision nodes reachable by d through a directed decision-free path
3:  for u in U and n in N do
4:    U' ← utilities belonging to the owner of n
```

5: $w \leftarrow$ all parents of n that are not descendants of d

6: **for** u' in U' **do**

7: **if directedEffectivePath**(n, u) and **frontDoorIndirectPath**(d, u', w) **then**

8: **return** true

9: **return** false

retract_edges(d)

1: $InfEdges \leftarrow$ all edges incoming to decision nodes in G

2: $removed \leftarrow false$

3: **for** (x, y) in $InfEdges$ **do**

4: $disabled((x, y)) \leftarrow true$

5: $D \leftarrow$ all decision nodes in G

6: **repeat**

7: $change \leftarrow false$

8: **for** d in D **do**

9: $Parents(d) \leftarrow$ parents of d

10: $Utilities(d) \leftarrow$ utilities of d

11: **for** p in $Parents(d)$ and u in $Utilities(d)$ **do**

12: $w(p, d) \leftarrow$ all parents of d except for p

13: **if** not **dSeparUseEnabled**$(p, u, w(p, d))$ **then**

14: $disabled((p, d)) \leftarrow false$

15: $change \leftarrow true; removed \leftarrow true$

16: **until** $change = false$

17: remove all disabled edges from G

18: **return** *removed*

directedDecisionFreePath(x_1, x_2)

return true if there is a directed, decision-free path from x_1 to x_2

directedEffectivePath(x_1, x_2)

return true if there is a directed path from x_1 to x_2 in which all decision nodes, except perhaps the first node of the path, are effective

effectivePath(x_1, x_2)

return true if there is an undirected path from x_1 to x_2 in which all decision nodes, except perhaps the first node of the path, are effective

directedEffectivePathNotThrough(x_1, x_2, Y)

return true if there is a directed effective path from x_1 to x_2 that does not go through any of the nodes in Y

backDoorPath(x_1, x_2, W)

return true if there is a back-door path from x_1 to x_2 that is not blocked by W

frontDoorIndirectPath(x_1, x_2, W)

return true if there is a non-directed front-door path with converging arrows at some node from x_1 to x_2 that is not blocked by W

dSepartUseEnabled(x_1, x_2, W)

return true if x_1 is d-separated from x_2 given W, by using only edges e having $disabled(e) = false$

A back door path in the above methods is defined as an undirected effective

path where the first edge comes into the first node, e.g., $< n_1 \leftarrow n_2 \ldots >$. A front door indirect path is an undirected effective path where the first edge comes out of the first node and, moreover, the path has converging arrows at some node, e.g., $< n_1 \rightarrow n_2 \ldots \rightarrow n_i \leftarrow \ldots >$.

5.3.1 Proof of correctness

We wish to show that the algorithm performs a legitimate simplification M' of the input MAID M, given our assumptions of how agents reason about available information. A simplification is legitimate if the equilibria of the simplified game are also equilibria of the original game, i.e., no new equilibria are introduced in the simplification. This allows us to solve the simplified game for an equilibrium with much smaller computational cost, and then use the solution for the more complex original game. On the other hand, a legitimate simplification is not guaranteed not to lose some equilibria of the original game (see Section 5.3.3 for an example showing how certain desirable equilibria may be lost in the simplification process). Besides legitimacy, the simplification generated by the algorithm is also shown to be "maximal," i.e, the maximum possible number of non-effective nodes is detected and removed. We also care about the effect of the order under which nodes are being eliminated, in order to guarantee that we do not have to identify a priori a particular "optimal order of elimination."

Definition 1. *A simplification M' of a MAID M is* legitimate *if all the Nash equilibria of M' are also Nash equilibria of M, in the sense that all nodes in M that have not been eliminated do not have an incentive to deviate from their equilibrium strategy in*

M' and all nodes that are not effective play in M according to a fully-mixed, uniform strategy.

We begin by first proving that, if the algorithm marks effective and non-effective nodes correctly, all Nash equilibria of the simplified MAID M' are also Nash equilibria of the original MAID. More precisely, since the original game also contains the decision nodes that were eliminated (and replaced by chance nodes) we need to show that extending the equilibria of M' by adding fully mixed uniform strategies for all non-motivated decisions yields a Nash equilibrium of M.

Theorem 1. *Let D be the decision nodes of the original MAID M and D' be the corresponding decision nodes in the simplified MAID M'. If σ' is a Nash equilibrium of M' then construct σ by adding fully mixed uniform strategies for all decision nodes in $D - D'$. Then σ is a Nash equilibrium of M.*

Proof. We prove this by contradiction. Let there be an agent with a decision node a who wishes to deviate from σ_a; there are two cases: either $a \in D'$ or $a \in D - D'$. In the first case, if the agent owning a wants to deviate to a strategy σ_a^1 in M then she would deviate to σ_a^1 in M' as well, contradicting our assumption that σ' is a Nash equilibrium in M'. In the second case, a was marked as non-effective, therefore a is not motivated, by Theorem 1. By the definition of a non-motivated node, $\mathbf{EU}^{<\sigma_{-a},d_1>}(a, \mathbf{q}) = \mathbf{EU}^{<\sigma_{-a},d_2>}(a, \mathbf{q})$ for every pair of actions d_1, d_2 of a and every configuration $\mathbf{q}$ of its informational parents. Therefore a provides with a fully mixed uniform strategy the same payoff as from any possible deviation from it. Therefore σ is a Nash equilibrium of M. The above reasoning holds even in cases where the agent of a owns other decision nodes besides a, from which he might try to simultaneously

deviate. The reason is that a is non-motivated due to strictly graphical properties of the MAID, not due to any particular parameters employed in other chance or decision nodes; thus strategies followed by any other agent—including the owner of a—elsewhere cannot cause it to become motivated. □

We then prove that the algorithm eliminates nodes correctly, irrespective of order. We do this first by looking at the four procedures that detect reasoning patterns. These work by explicitly following the definitions of the four reasoning patterns, so they are correct (details are omitted). We then look at the first phase of the algorithm (lines 6-18). First, however, we need the following lemma.

Lemma 1. *If R_n^E is the set of reasoning patterns that hold for a decision node n when the set of edges in the MAID is E, then $R_n^{E'} \subseteq R_n^E$ for all $E' \subseteq E$.*

Proof. Consider a MAID with edges E and a set R_n^E of reasoning patterns holding for decision node n. Now remove an edge $e \in E$, such that the MAID now has edges $E' = E - \{e\}$. Suppose now, for the sake of contradiction, that $R_n^{E'} \not\subseteq R_n^E$. This means that a reasoning pattern r did not exist under E but exists under E'. Take the paths P_r of this reasoning pattern and let $E(P_r)$ be the set of their edges. Clearly, $E(P_r) \subseteq E' \subset E$ so all the paths the reasoning pattern r depends on existed in the original MAID with edges E. Thus the only reason why r did not hold under E was that one or more of its paths were blocked at some node. Let $p \in P_r$ be one such path with blocking set W_p ($W_p = \emptyset$ if the definition of the reasoning pattern required no d-separation properties to hold for that path) and let b be the node where p was blocked by W_p. If p has non-converging arrows in b then $b \in W_p$, so p should be blocked again under E', since no nodes were removed, only an edge. If p has converging arrows in

b then it means that neither b nor any of its descendants were in W_p. But removing e can neither add b to W_p, nor cause the set of its descendants to grow. Therefore, under E', too, W_p will block p at b and therefore our argument is contradictory. □

Lemma 2. *If a node is identified in some identification phase under some order, it will be so identified under any order.*

Proof. Let $n_1, ..., n_k$ be an order of identifying non-effective nodes and $n'_1, ..., n'_k$ be a different order. Also define as E_n the set of edges in the MAID after node n has been eliminated and as E the edges in the MAID in the beginning, before any elimination takes place. Suppose that under the new order node n_1 is placed at position h. Then, by Lemma 1, in the identification phase and under the new order n_1 will be eliminated irrespective of h, since $E'_h \subseteq E$ and node n_1 was eliminated under E. We then reason by induction. Now assume that $n_1, ..., n_i$ have been eliminated. Then n_{i+1} will be eliminated at the latest in the next phase after all of $n_1, ..., n_i$ have been eliminated, because the set of edges present will be a subset of E_i. □

For the second phase of the algorithm (pruning), this is exactly implemented as in [83], which contains the proof of its correctness.

Theorem 2. *The algorithm produces a correct and maximal simplification of a MAID.*

Proof. We know that the operation of each phase is correct. Moreover, we know the pruning phase only removes edges. Thus, by Lemma 1, it does not matter in the context of the identification phase on which iteration of the algorithm the pruning phase removes an edge e, as long as it eventually removes it (on some iteration). Thus

the only thing we need to establish is that no operation in the identification phase might ever prevent an edge from being removed in the pruning phase.

The pruning phase works by testing for certain d-separation properties, while the identification phase only removes edges (never adds). Thus, in spirit similar to the proof of Lemma 1, if the MAID has edges E in the beginning of the pruning phase and an edge $e = (x, y)$ is removed during its execution, then if the MAID had edges $E' \subset E$ then e would still be removed. If e was removed with edges E then x was d-separated from the utility nodes of y given $\{y\} \cup \{d : (d, y) \in E, d \neq x\}$, by definition of [83]'s algorithm. Now under the smaller set of edges E' it is the case that x must, again, be d-separated from y's utility nodes, since the removal of any edge in $E - E'$ cannot have made these two less separated. Therefore, no matter in which order we execute the two phases and no matter what their intermediate results are, the end product is the same.

Furthermore, the iteration of identification and pruning phases will terminate. Neither adds an edge and there are at most E edges to remove, so the process will eventually terminate. □

We have shown that the algorithm's operations are consistent with our assumptions and that it will always return a maximally simplified MAID. In the following section its complexity is analyzed.

5.3.2 Algorithm Complexity

The complexity of the algorithm is easy to estimate. We first begin with the bottom-level procedures which are path operations. Those that do not involve a non-

empty blocking set are instances of graph reachability, which can be performed in $O(E+N) = O(E)$ time, assuming $E > N$. If the blocking set is non-empty, however, every time a path is expanded one needs to check that it is not blocked at that node. This can be done using an algorithm such as BayesBall [116], which is $O(E)$.

We also use *memoization* to improve the computation of blocking properties along the paths. In particular, whenever we query whether a path with converging arrows at at a node b is blocked by a set W, we store the result $blocked(b, W)$ in a hash table. Subsequent queries for the same node and blocking set first check the hash table for an already computed result and only execute the full operation (costing $O(E)$) if needed. After each iteration, since the structure of the graph has changed, we drop all memoized entries (line 16).

Theorem 3. *The algorithm simplifies the MAID in time $O(D^2N^2E)$, where D is the number of decision nodes in the graph.*

Proof. We have established that all path operations take polynomial time. In particular, suppose C and U are the number of chance and utility nodes. Then procedure **df** performs $O(U)$ simple path operations, so it costs $O(UE)$ in the worst case. Manipulation (**man**) performs $O(DU^2)$ simple operations, for a total cost of $O(DEU^2)$. Signaling (**sig**) requires certain paths to satisfy blocking properties and performs $O(CDU^2)$ of those, for a total worst-case cost of $O(CDU^2E^2)$. Here the E^2 results from the following: For every combination of nodes related to the signaling patterns we need to find certain paths with graph reachability ($O(E)$); for each such path, at every step we need to check for blocking properties, which adds another $O(E)$, for at total of $O(E^2)$. Finally, revealing-denying (**rev**) performs $O(DU^2)$ blocking-sensitive

path operations and thus has worst case cost of $O(DU^2E^2)$.

We see that signaling is the most expensive of these operations. However, with memoization, its worst-case complexity can be reduced. We reason as follows. There can be a total of $O(DN)$ blocking sets required for the purposes of identifying reasoning patterns in the graph (for every decision node there can be one blocking set including all of its parents but one, and there are $O(N)$ parents per decision). Thus, even if we were to calculate blocking properties for all nodes and possible sets, the time per iteration would be bounded by $O(DN^2E)$. In a similar fashion, the time complexity for revealing-denying identification can be bounded.

The algorithm as a whole also performs at most $O(D)$ iterations in the outer loop. This is because if two consecutive iterations eliminate no nodes the algorithm by definition terminates, since the pruning phase of the second iteration will remove no edges. Therefore the total cost of the algorithm is polynomial and on the order of $O(D^2N^2E)$. □

Of course it has to be noted that the expected performance of the algorithm is likely to be much better than its worst-case bound calculated above. In particular, the evaluation of the if structure in line 10 is short-circuited, meaning the expensive **sig** operation is only evaluated if **df** and **man** are false, since it is sufficient to show that a node participates in one reasoning pattern to be effective. Moreover, real games very likely have much fewer reasoning patterns mainly consisting of direct effects and manipulations, especially after the pruning phases of the first iterations have reduced the number of edges in the graph that are important to signaling and revealing-denying patterns.

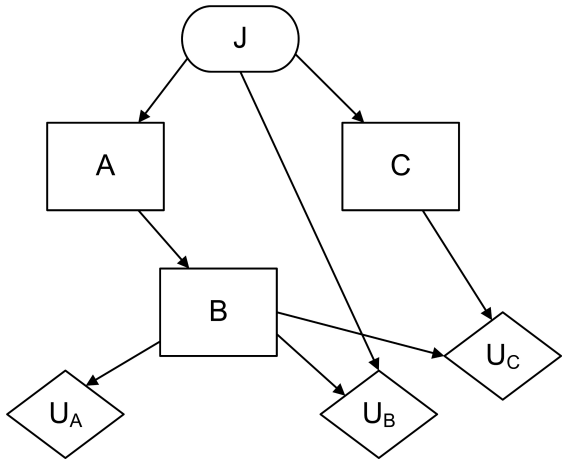

Figure 5.12: MAID representing the full example game

5.3.3 Time Savings in Equilibrium Computation

Consider an example game, represented as a MAID in Fig 5.12. We have agents A, B and C. Agent A draws a card J, whose value can either be H, M or L. Only agents A and C have knowledge of that card, yet A may communicate its value to agent B, not necessarily truthfully. B gains \$30 by guessing the value of the card correctly. A gains \$10, \$5 or \$1 if B guesses H, M or L, respectively, no matter what the real value of the card is. C, on the other hand, gains \$30 if his choice of H, M or L differs from that of B.

This game has an extensive game form representation with 3^4 leaves. On the other hand, the algorithm can discover the following subtleties in the above scenario: Agent C's decision is not affected by the card value, so the information arc (J, C) can be removed. Then, the decision node for A is non-effective, i.e. A has no reason to act differently upon seeing any of the card values. Knowing that, B will ignore what A tells her and just randomize equally between H, M and L. Thus the large

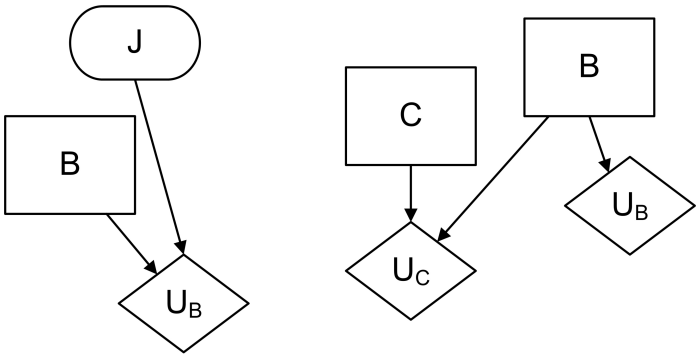

Figure 5.13: Simplified MAIDs

game is reduced to two mini-games of 3^2 leaves each (Figure 5.13). Adding more players to the game, one can easily show that computational savings are exponential in the number of agents, whereas the algorithm that breaks the game down runs in polynomial time.

Of course this process is not without loss. Removing the redundant edges following the algorithm of [83] happens to eliminate certain, possibly efficient, equilibria. For example in our little scenario an equilibrium could be as follows: A always communicates H to B, B always believes him and C randomizes equally between M and L. This is clearly a Nash equilibrium: if B always believes him, then A responds best to that by communicating always H to her. Knowing that A will always communicate H and B will believe him and respond with H as well, C will choose either M or L to win. Finally, since the card is H with probability $\frac{1}{3}$ no matter what A tells her, B's strategy to always believe A is equally good as pure randomization. In this equilibrium, the players' expected payoffs are $\$(10, 10, 20)$, whereas in our reduced game equilibrium all players will randomize equally, yielding expected payoffs $\$(\frac{17}{3}, 10, 20)$.

One can see that the equilibrium that was "lost" is a Pareto-improvement upon the one that has been retained.

Chapter 6

The Reasoning Patterns: Helping Humans

In the previous chapter, Pfeffer & Gal's theory of reasoning patterns was used to assist game modelers in representing, studying and solving complex games. In this chapter, the theory is also demonstrated to be useful to human decision-makers. An empirical study is presented, in which people were asked to play a repeated Bayesian game in pairs. Half the participants had access to a computer assistant that gave them advice on how to play the game, while the remaining half did not have access to it. The computer assistant used the game's reasoning patterns to generate arguments for or against particular strategic choices the subjects made, and quantified the effects of their actions. The group of subjects who had access to the assistant performed significantly better than the second group. This result suggests that the reasoning patters can be used to offer non-trivial insights to people about how to make decisions in complex games.

6.1 Using Reasoning Patterns to Assist Human Decision-Makers

People are often expected to make decisions in complex environments. In such domains, it may be hard for them to identify the optimal decision, or even a reasonably good one, due to the size of the domain, limited computational resources, or the need to decide under strict time limits.

How can computers aid people in such decision-making contexts? A straightforward approach would consist in computing the optimal decision or strategy—for instance, a Nash equilibrium of the game—and advise people to implement it. This approach, however, suffers from a number of limitations. First, Nash equilibria are computationally very challenging to compute, and require exponential time in the general case [28]. Second, for most games of interest, such as repeated games or games of incomplete information, there is a multitude of equilibria, and it is hard to know which one to suggest to the human decision-maker. Third, a Nash equilibrium is optimal with respect to ideal "rational" play, but not necessarily with respect to the actual play of the other players, especially other humans'. In a tournament in which agents had to play a repeated version of the well-known 'rock-paper-scissors' game, agents implementing the Nash equilibrium of the game came in the middle of the pack [13]. A computer suggesting this equilibrium strategy for rock-paper-scissors would not be offering very useful advice.

The most fundamental problem with this approach, however, is that it takes people out of the decision making process. If a human decision maker bears ultimate

responsibility for the outcome of the decision, he or she may want to understand the reasons behind the decision and personally make it. In a recent study [21], researchers recommended the Nash equilibrium strategy to subjects in a two-player game. They found that subjects were reluctant to follow the recommendations, even when they knew that the other player was recommended the same unique equilibrium. This experiment represents ideal conditions, as in ordinary games players cannot be sure that others are following the same equilibrium strategy, or even an equilibrium strategy at all, so they will plausibly be even less likely to entrust their decision to an equilibrium calculator.

Furthermore, humans have judgment, insight and intuition that are not available to the computer in its equilibrium analysis. Computers are capable of fast execution of algorithms, which allows them to excel in quantitative analyses. In games, defined as sets of agents, strategies and utility functions, a computer will be able to evaluate probability distributions over outcomes, expectations over the players' utilities, and employ numerous techniques to help them maximize their payoffs. On the other hand, humans have good intuitions as to what models or strategies their opponents are most likely to construct and follow. Because people often have reasonably accurate beliefs about how other people tend to behave, they gain the edge over computers in identifying what is "reasonable." In addition, humans may be able to cut through complex games to identify key patterns and simplifications that lead to good strategies.

In this section the theory of reasoning patterns is used to generate advice and present arguments in favor of different strategies to the human decision-maker. These arguments are derived directly from the reasoning patterns of the game, and it helps

people understand, for each action, what will be accomplished by it, how it will affect other agents' choices in the future, how it is going to benefit them, and what the potential risks are.

It is hypothesized that people may benefit from such advice in two key ways. First, it relieves them of having to *quantify* the effects of their actions. The advice provides numeric estimates of the benefits and risks of particular strategies. People can easily compare the numbers and quickly identify which course of action best serves their goals and preferences. Second, the advice may contain insights about the game that people had not thought of at all. This is particularly the case in more complex games, in which it is harder to identify all the relevant arguments for and against each strategy.

This hypothesis was tested empirically in a complex repeated game of incomplete information. Subjects were asked to play this game in randomly chosen pairs. Half of them were given advice generated by the game's reasoning patterns, whereas the other half played the game with intuition alone. It was observed that, on average, subjects who received the advice performed significantly better compared to those who did not have access to it. These results demonstrate that the reasoning patterns can be a valuable source of intuition for decision-makers, and that they can improve people's ability to navigate and understand how to play complex games successfully.

6.1.1 The principal-agent game

To test the hypothesis that arguments provided to humans as advice are beneficial, a simple, yet subtle game was constructed, which we will refer to as the *principal-agent*

(p-a) game, due to its resemblance with similar games in the economics literature [47]. In those games, one player, called the "principal," must choose whether to hire, and how much to pay, another player, called the "agent."[1] The agent has a *type*, which is hidden from the principal. The agent's choice consists of selecting an effort level. Both players' payoffs depend on the actions of the agent (assuming she was hired) and the payment exchanged. This generic model is of interest, because it may serve as a good approximation for many real-life situations, such as hiring employees, constructing incentive-compatible payment schemes, and teaching students, among others.

The particular version of the principal-agent game used in this experiment follows a similar structure. The principal and the agent are both initially given some resources, which are privately known to them. At first, the principal may choose to transfer part (or all) of his resources to the agent, or he may choose to transfer nothing. Then the agent may opt to spend some of her resources to move towards a goal square on a grid.[2] The principal's payoff is increasing in the amount of resource held in the end of the game, as well as the closeness of the agent to the goal. The agent's payoff is, however, dependent upon her "type." Agents can either be "resource-lovers," who value resources but not distance from the goal, or "goal-lovers," who value closeness to the goal but not resources. Clearly, the incentives of goal-lovers are aligned with those of principals, as they will naturally try to move as close to the goal as possible, which increases principals' utility. On the contrary, the incentives of resource-lovers

[1]The term "agent" is used here not in the common sense of a computer agent, but in the narrow sense in this particular class of games in economics. We shall refer to both parties, the principal and the agent, as players to avoid confusion.

[2]Throughout the chapter we will refer to the principal and the agent with masculine and feminine pronouns, respectively.

are not, as they will not waste any of their resources to move towards the goal square, something that carries no value for them.

The repeated version of the above game is used, however, in which both types' incentives are less clear-cut. In each *round*, the game is reset, and both players' resources are restored, but the same principal is paired again with the same agent, whose type remains unchanged. This allows for nuanced behavior to form: First, the principal can maintain a belief over the agent's type and update it after every round, by observing her actions. Second, the agent may find it beneficial to adjust her behavior in order to reveal or mask her true type, such that the principal may take actions in subsequent rounds that improve her payoff. In other words, the agent may choose to manipulate the principal's beliefs. For instance, a resource-loving agent might still choose to move towards the goal just by a few squares. Although this reduces her payoff in the current round of the game, it prevents the principal from inferring her true type and thus denying them any resources in all subsequent rounds, which would greatly reduce her payoff. (Because her initial resources held by the agent are known only to her, not moving all the way to the goal can be attributed to her lacking the necessary resources to reach the goal square.)

Why was this particular game chosen? There are many reasons: For one, it is difficult to solve it analytically. There is no simple way to model the principal's belief update function, which maps current beliefs and observed actions to updated beliefs. This is because the probability of an agent taking an action does not just depend on her type and her (privately known) resources, but also on the degree in (and direction to) which she has chosen to manipulate the principal's beliefs in

this round. A similar argument holds for the agent: to be able to reason about how the principal will interpret any one of her actions, she must be aware of the model the principal has constructed of her strategy. Technically, this is a Bayesian repeated game, in which types are constant throughout all stage games, and each stage game has imperfect information. An impossibility result has been proved for Bayesian learning in such games [89]. Moreover, since this is a repeated game, one would expect a multitude of equilibria, alluded to by the folk theorems. In addition, people's play in this game might deviate significantly from equilibrium. Thus the traditional game theoretic approach suffers from the disadvantages outlined above: equilibria are hard to compute, numerous, and it is not clear that they are relevant.

Yet, despite its complexity, the principal-agent game has strong appeal. Although optimal strategies are difficult to compute, something can be said of what might constitute a "good" or "reasonable" strategy. First, both players' scores in any given round depend directly on their actions: for the principal, a larger transfer is costlier, and for the agent the move affects her score either by influencing distance (for the goal-lover) or resources possessed (for the resource-lover). Second, the goal-lover agent has an incentive to move in a way that reveal her type more explicitly to the principal. This is because, if the principal becomes convinced that the agent is a goal-lover sooner, he might transfer larger amounts of resource in more future rounds. Conversely, the resource-lover has an incentive to hide her type, by not performing moves that are "characteristic" of resource-lovers (e.g., not moving), so as to maintain "useful doubt" in the principal's mind. Therefore, the game is a good example of what is *technically hard* but perhaps *intuitively approachable*.

6.1.2 Experiment implementation

For the experiment, the principal-agent game was implemented in the Colored Trails (CT) framework [53], a client-server architecture system for studying multi-agent decision making. CT is, in the simplest sense, a board game played over a computer network. In the experiment there was a 5×5 board of colored squares (see Figure 6.1). The squares were uniformly and independently colored in each round with a palette of four colors (red, green, orange and purple). There were two players, the principal and the agent. The agent was either a "goal-lover" or a "resource-lover" with probability 0.5. The agent was positioned in the lower-left corner (coordinates (4,0)). At the same time, a goal square was always placed in the upper-right corner (0,4). Both players in each round were given five chips at random (i.e., any combination of colors adding up to 5 chips was equally likely), and each player could only see his/her own chips, not his/her opponent's. These chips could be used to move on the board: for the agent to move on a red square, for example, she had to give up a red chip.

Each round was then played in two phases: in the *transfer* phase, lasting 2 minutes, the principal could initiate a transfer of chips. The principal was allowed to transfer any subset of his/her chips, from the null transfer to everything. As soon as the transfer was carried out (or the 2 minutes expired), the round moved on to the *movement* phase, in which the agent could drag her icon on the board and move it (no diagonal moves were allowed), spending chips in the process. After the movement phase the round was ended. Players would get scores based on their actions (in a so-called *score update* phase), and a new round would begin between the same principal and the same agent with probability 90%. With the remaining 10% the game would

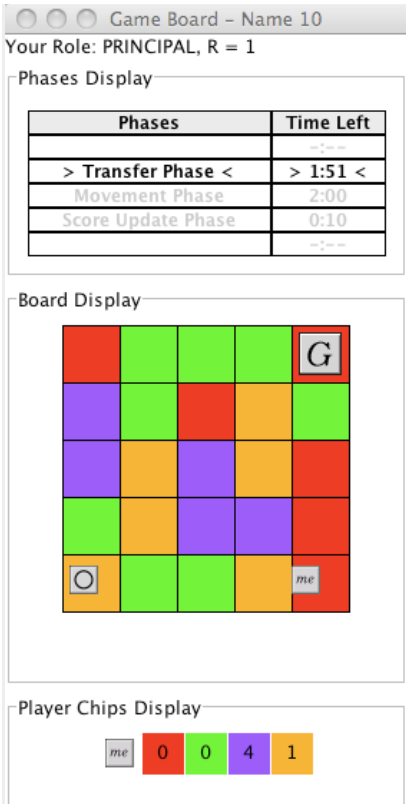

Figure 6.1: The user interface of the CT game

end and, when all games between all subjects ended, the subjects were again randomly re-paired. Thus, for example, a subject who was a principal in the first game might assume the role of a goal-lover agent in the second game. Subjects were fully aware that the identity of their opponent would change between games, but not between rounds of the same game.

More specifically, the principal's scoring function was $\pi_p(c_p, c_t, x) = 50(|c_p| - |c_t|) + 65(8 - dist(x, G))$, where x is the position the agent has chosen to move to, c_p and c_t are the initial chipset given to the principal and the transfer made, $|\cdot|$ denotes set size, G is the goal position and $dist(\cdot, \cdot)$ measures the Manhattan distance between two points on the board. The scoring functions for the goal-lover agent was $\pi_g(x) = 20(8 - dist(x, G)) + 250 \cdot \mathrm{I}[x = G]$, where $\mathrm{I}[\cdot]$ is the indicator function. Finally the resource-lover gained $\pi_c(c_a, c_t, p) = 20(|c_a| + |c_t| - chips(p))$, where c_a is the agent's original chipset and $chips(p)$ denotes the chipset required to move from the original position along path p. All scoring functions were common knowledge to the subject pool.

Subjects were paid with real money according to their scores, as compared to those of other players. Scores were normalized within a single type, such that a subject who happened to play as a principal and a resource-lover was paid as a function of how high his/her score was compared to the average principal and the average resource-lover. This was done to eliminate unfairness in payments, as principals generally had higher absolute numeric scores than agents, so if a subject happened to assume the agent role she would be paid less for no reason. Proper procedures were followed for teaching subjects how to play the game, letting them play a few test rounds,

answering their questions, and debriefing them in the end.

6.1.3 Using reasoning patterns for advice generation

Reasoning patterns are naturally appealing for generating meaningful, intuitive arguments for people on how to play a game. The main advantage of reasoning patterns is that, while they are deeply grounded in theory and easy to discover and quantify by a computer, at the same time they are describable in a way that people may find intuitive, For example, a "manipulation" reasoning pattern might be described as "if you do x then this other player will want to choose y, which will boost your utility by k points."

Identifying the reasoning patterns in a game might be easy. The more challenging step is to transform them into arguments. The main idea here was that every reasoning pattern r for decision D of a player can be represented with a scoring function $v_r : A(D) \to \mathbb{R}$, where $A(D)$ is the set of actions available to the player in D. Intuitively, higher values of v_r mean better choices. This scoring function is constructed to be *localized* in nature, as it ignores portions of the game that lie outside its main description. Technically, the portion of the MAID graph that lies outside the paths forming the reasoning pattern is ignored. The format of the scoring functions used for each of the patterns is as follows:

- *Direct effect:* We score an action with respect to direct effect in decision D by summing over the expected payoff the player stands to make in all the utility nodes within the MAID that descend from D, and which are not blocked by other players' decision nodes. This measures exactly captures direct effect,

reflecting what the player can accomplish by acting on her own, without depending on others' decisions.

- *Manipulation:* The scoring function for manipulation (of player A to player B) measures the expected increase in utility obtained due to B taking an action because she has observed A's decision. This increase is usually computed with respect to a reference action of A, which can be chosen arbitrarily, since it serves only for comparison purposes.

- *Signaling:* When A signals C to B we compute the extra utility obtained by A by causing B to update his probability distribution over C, and thus change the probability of his actions that affect A's utility.

- *Revealing-denying:* Similarly, we score an action with respect to revealing-denying by computing the incremental utility the player obtains by causing another's belief to be updated in a particular way.

Each such scoring function represents an argument. Different arguments may be computed for a particular decision. Thus, if there are two reasoning patterns r_1 and r_2 of D, it might be the case that a particular action $a \in A(d)$ fares well in v_{r_1} but poorly in v_{r_2}. This is because a might be a good action in the portion of the game captured by r_1, but not so good in the portion described by r_2. Note also that the total utility to the agent of taking a is not necessarily the sum $v_{r_1}(a) + v_{r_2}(a)$, as the MAID portions of r_1 and r_2 might be overlapping.

6.1.4 Reasoning patterns in the p-a game

The MAID for the multiple-round p-a game is constructed by connecting copies of the corresponding MAID for a single round. The MAID for the first two rounds of the p-a game, for instance, is shown in Figure 6.2. In each round i, there are two decision nodes, P_i, the principal's choice of a chip transfer, and A_i, the agent's choice of a movement path. There are also four chance nodes, B_i, the board coloring (known to both players), C_i^p and C_i^a, the principal's and the agent's chipsets (known to their respective players), and T, the agent's type. Notice that the agent's type is not subscripted, because it remains the same throughout the game. Also, variable T is only observable to the agent, not the principal. Between rounds there are two types of arrows: *(i)* observation arrows, e.g., an arrow from A_{i-1} to P_i, meaning that the principal in round i has observed the agent's action in the previous round, and *(ii)* no-forgetting (perfect recall) arrows, e.g., from P_{i-1} to P_i, meaning that the principal has not forgotten his action in the past.

This game has the following reasoning patterns:

1. *Direct effect for P_i:* For all rounds i, the principal's decision P_i has direct effect, meaning that his choice of a transfer affects his score in round i. This is because the number of chips he chooses to transfer directly affects his score.

2. *Direct effect for A_i:* For all rounds i, again, the agent's decision A_i has direct effect, as her movement affects her score directly, e.g., if she is a resource-lover, spending chips to move closer to the goal decreases her current-round score, whereas if she is a goal-lover it increases her score.

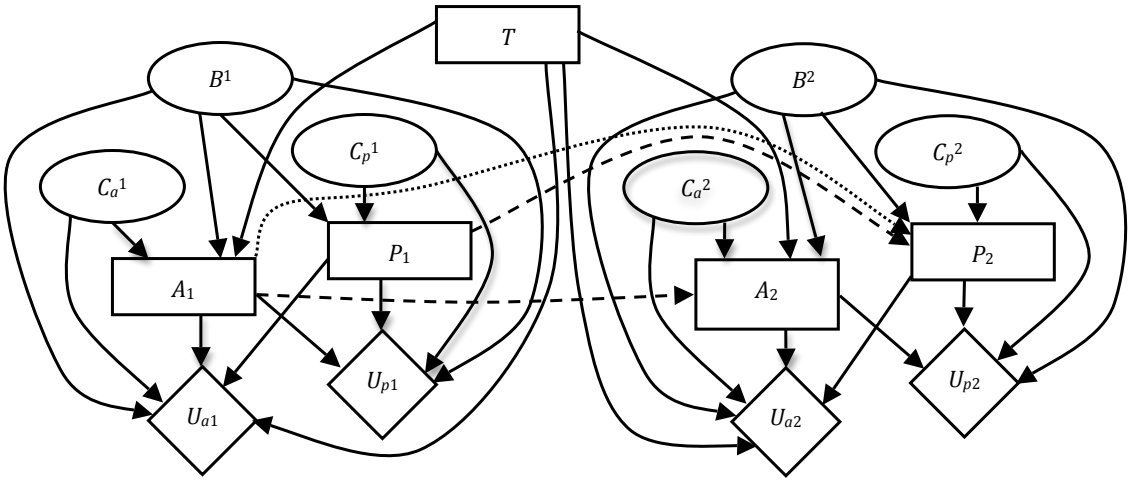

Figure 6.2: Two rounds of the p-a game

3. *Manipulation for P_i to A_i:* For all rounds i, the principal influences the agent through manipulation. This captures the fact that, if the principal transfers more chips to a goal-lover, the principal may allow her to move closer to the goal, and thus positively affect the principal's utility, which is decreasing in the distance between the agent and the goal at the end of the game.

4. *Signaling for A_i to P_{i+1}:* In every round, the agent takes an action that the principal will observe and update his belief over her type. His chip transfer in the following round will then be decided under this updated belief and may be crucially affected by it. The agent has an incentive to convince the principal that she is a goal-lover, as it is beneficial for the principal to transfer more chips to the agent when he believes she is a goal-lover. An actual goal-loving agent may thus wish to move as close to the goal as possible to advertise her type, while a resource-lover may want to avoid total immobility, to maintain some

"useful doubt" in the principal's mind.

5. *Revealing for P_i to P_{i+1}:* In every round the principal's transfer serves two purposes: On the one hand, it enables the agent to move closer to the goal in that same round. On the other hand, it is a useful "exploration tool." In particular, by making a transfer and observing the agent's response, the principal updates his belief over her type. Some transfers are more helpful than others in that latter respect. For instance, if the principal transferred all his chips to the agent in one round, the agent would have no excuse for not moving all the way to the goal square. If she would refrain from doing so, this would reveal to the principal that she is a resource-lover with very high probability. We say that "the principal reveals the agent's type to himself in the future."

To generate advice for the p-a game, these five reasoning patterns applicable to every round of the game are used, as well as the scoring functions defined for each pattern. In particular, we compute $v_{r_i}(\cdot)$ for every pattern r_i of the principal (agent), and we output these values to the human subject acting as the principal (agent), along with natural language text explaining what these numbers mean. In this advice there is no direct suggestion of a specific optimal (or good) action the player must take. The point here is to *educate* the subjects with the intuition gained by examining the reasoning patterns, and then let them take over the task of meaningfully weighing them, combining them and reaching a final decision.

Both players may seek advice while playing the game, but only for a particular action (transfer or move). For example, the principal can use the advisor interface to ask "what do you think about a transfer of 1 blue chip and 2 green?" In particular, if

players wish to receive advice, they turn on the "advice mode" by pressing a button on the game's interface and then choosing a specific move. Instead of the move being carried out, advice is generated regarding the move and presented to the player.

The advice for the principal consists of a general and a transfer-specific part. The general part explains the three relevant reasoning patterns (1, 3 and 5 above). In this general part the advisor also computes an estimate of the likelihood that the agent is a goal-lover, giver her actions so far. The specific part of the advice mentions (*i*) the cost of the transfer, (*ii*) the expected gain in points due to the agent moving because of the transferred chips (compared to making a null transfer), and (*iii*) the extra points expected to be gained in the next round due to a more refined distribution over the agent's type (exploration). These three quantities correspond to the scoring functions associated with the three reasoning patterns of the principal's decision.

Similarly, the agent receives general-purpose advice, which explains that her move will be used by the principal to reason about her type. The advice also mentions that a resource-lover should try to use at least the chips transferred to her to the extent that the board color configuration allows. The move-specific advice mentions: (*i*) the score obtained in that round by making that particular move, and (*ii*) the extra points expected to be gained in the next round due to controlling the principal's belief with that move (compared to not moving). Again, these two correspond to the two reasoning patterns associated with the agent's decision in that round.

6.1.5 Strategic assumptions and approximations

As mentioned earlier, the scoring function for each reasoning pattern only looks at the MAID subgraph lying within the paths comprising that reasoning pattern. However, the utility obtained by an agent within that subgraph may be affected by what other agents might be choosing outside the subgraph. We therefore need to make assumptions about the strategic choices these agents are making in that other part of the MAID. The same is true for belief updates. For example, if we wish to update the principal's beliefs about the agent's type based on the agent's actions, we need to know what strategy the agent would use for each type. We argued earlier that in the p-a game this is difficult to compute, as the agent might be actively trying to mask her type.

To address these issues, we plug in some approximations for players' strategies. In particular, we define $\hat{\sigma}_p$, $\hat{\sigma}_a$ for the principal and the agent, respectively, to be *myopic quantal response strategies*, and use those whenever we need to make calculations using the other agent's strategy, either in the section of the MAID that lies outside a reasoning pattern or to perform belief updates. This involves two assumptions: (1) the players myopically only consider their utility in the current round when considering their play; and (2) they implement a quantal response. In a quantal response equilibrium [82], a player determines the utility $\pi(a)$ for each action a, given his beliefs about the other players' strategies. The player then chooses action a with probability $\frac{e^{\lambda\pi(a)}}{\sum_{a'} e^{\lambda\pi(a')}}$, where λ is a parameter. As $\lambda \to \infty$, the player is completely rational and chooses his best response with probability 1. If $\lambda = 0$ the player chooses actions uniformly at random. Thus by choosing λ we can control to what degree

agents choose their best scoring action. We set $\lambda = 0.5$ in the above formula, because it generated decision-making behavior matching the actual observed behavior of people in a preliminary test run of the experiment.

One might argue that the advice given is only as good as the assumptions made about strategies used in computing the scoring functions. This is certainly true. However, no great effort was made to be particularly clever or realistic about the strategies, yet nevertheless solid results were obtained. This combination of myopic strategies (or at least strategies with short look ahead) and quantal response may work well in many games. Quantal response is important because it smooths out any errors made in the myopic assumption. For example, myopic best response would tell a resource-loving agent never to move. Quantal response smooths this out so that the resource-loving agent moves, at least by a small amount, a good fraction of the time. This is important because then, if the principal observes that the agent moved a little bit, he can still place high probability on the agent being a resource-lover. If the principal believed the agent was purely myopic, then after the agent moved a little bit the principal would believe she is a goal-lover with certainty.

It must also be clarified that, whenever a subject asked for advice, the algorithm had to perform several computations, including updating probability distributions and calculating expectations of values. Whenever possible, these computations were performed by exact inference or closed-form solutions. In cases where that was infeasible, however, sampling approximations were instead used. For instance, to compute the expected number of points the principal could expect when dealing with an agent who was believed to be a goal-lover with some probability, the algorithm sampled

several board color configurations, principal and agent chipsets for the next round, and then responses from both players according to $\hat{\sigma}_p$ and $\hat{\sigma}_a$, and then computed payoffs.

6.1.6 Results

The experiment was performed with 18 subjects in two sessions (one with 10 and one with 8 subjects). Half the subjects in each session had access to the advice, while the remaining half did not, but were aware of some of their opponents being able to access it. Subjects were randomly paired and assumed different roles between games, but advice-seeking was either on or off for a particular subject throughout the experiment.

A subject's performance was calculated as follows: Suppose $R = \{p, g, c\}$, where p, g, c stand for "principal," "goal-lover," and "resource-lover." If subject i assumed roles $R_i = \{p, c\}$ during the experiment, and made an average π_i^p and π_i^c points respectively, his score is defined as $s_i = \frac{1}{2}(\frac{\pi_i^p}{\bar{\pi}^p} + \frac{\pi_i^c}{\bar{\pi}^c})$, where $\bar{\pi}^t$ is the average score of all players of type t. In general, the performance of a player i is defined as

$$s_i = \frac{1}{|R_i|} \sum_{t \in R_i} \frac{\pi_i^t}{\bar{\pi}^t}$$

Similarly, the performance of a group G containing the set of agents $A(G)$ is defined as

$$s_G = \frac{1}{|A(G)|} \sum_{i \in A(G)} s_i$$

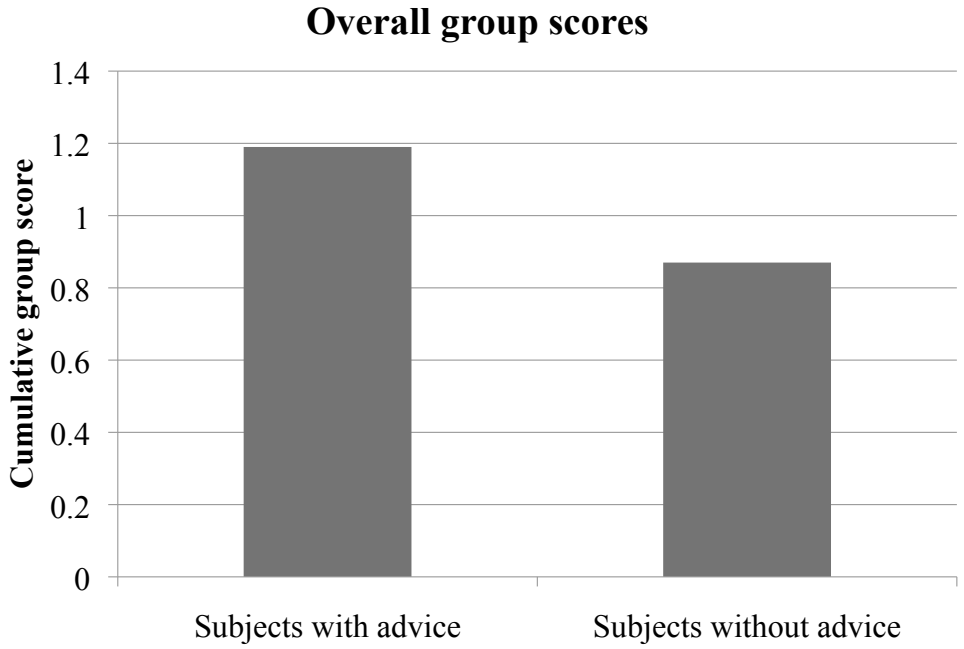

Figure 6.3: Cumulative scores in the p-a game

Then let A, A^C be the two groups of subjects, those playing with advice and those not having access to it. The experiment measured that $s_A = 1.19$ and $s_{A^C} = 0.87$, a performance boost equal to 37% from taking advice (see Figure 6.3). This result is statistically significant at the 1% level.

One other interesting result is that, if the payoffs of players in A^C are examined, but two cases are taken: when the player was paired with someone also in A^C and when the player was paired with someone in A, it can be observed that in the first case subjects achieve an average of 11% higher scores (see Figure 6.4). This implies that a subject is always better off using the advice, regardless of whether his/her opponent also has access to it.

One could also examine the "raw" in-game scores subjects attained, instead of

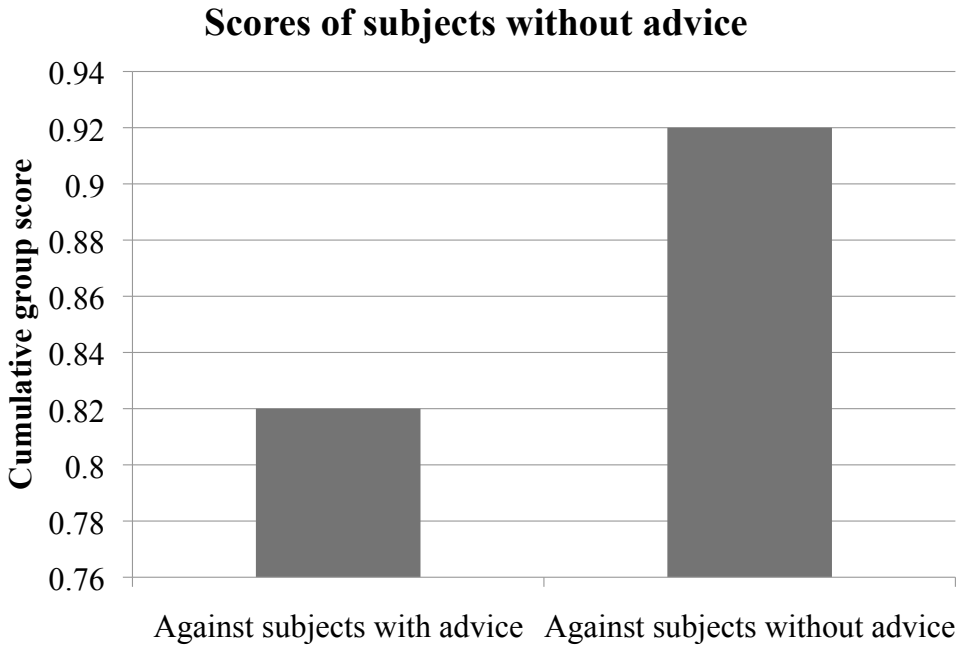

Figure 6.4: Cumulative scores of agents without advice

their normalized payoffs. This serves to answer whether total payoff (for both agents) increases. (Observe that, if the advice causes both players' payoffs to increase by, say, 20%, then this will not be reflected in the normalized scores used above.) However, social welfare, defined this way, generally does not increase among subjects using the advice (see Table 6.1).

6.2 Discussion

The results show that people who are given advice in the form of arguments based on the reasoning patterns did better than people who were not given advice. There are three main questions that need to be discussed with respect to these results. First,

	principal	goal-lover	resource-lover
A against A	297.5	20.5	130
A against A^C	605	60.5	150
A^C against A	297.5	140	100
A^C against A^C	295	60	130

Table 6.1: Average (raw) scores in the p-a game

can the performance gap be attributed solely to the quality of the advice? Second, how do these results generalize to other games or situations? And third, how would this advice compare to direct instructions, i.e., outright saying to people what actions to take?

The first question is important if one considers possible side-effects of the advice-giving mechanism. For example, one might claim that, for subjects that could request advice, just the fact that they spent more time on their decisions—even if the "content" of the advice was not helpful—improved their performance. Alternatively, the gap might be explained by user interface modifications (e.g., that the advice-giving interface was in some sense "smart" and led subjects to better actions, without the advice providing them any insights per se). To ensure as much as possible that the GUI did not interfere, the simplest possible form of UI was used, a bare popup window with just text, no graphics and no explicit interaction within the advice window. The text contained the general-purpose advice in one paragraph, followed by the transfer- (or move-) specific advice, in one sentence per relevant reasoning pattern. The phase time limit was also kept to 2 minutes for subjects who got the advice, so that any potential benefit from spending more effort in the decision-making process would be outweighed by less time to actually make the decision.

As for the second concern, this first experiment is to be treated as a first step in exploring the possibilities of the method. Although no claim can be made of its universality, I am hopeful that its usefulness is not restricted to the principal-agent game of the experiment, but is extensible—albeit not effortlessly—to other games where arguments can be similarly identified. It is the deferred to future work to further develop and formalize the technique for quantifying and combining reasoning patterns to generate more nuanced advice and perhaps identify good and intuitive strategies automatically, without the need for a human to weigh each argument against other, possibly conflicting arguments, in the advice.

Finally, we need to consider other means of generating advice, especially direct instructions. Directly telling people what actions to take is, after all, a lot simpler than giving them insights about the game or evaluating candidate actions only after they suggest them, as in the experiment. Arguably, in games with a single optimal strategy which can be easily computed, instructing people to follow it seems preferable to relying on their exploration and understanding. The advice-generation technique is more suited, however, for games in which there is no single optimal strategy, or running an optimization algorithm is computationally very expensive, or the strategy space of the game is too big to explore. It is in those complex games that the merit of the reasoning patterns is most pronounced, since they allow for qualitative arguments to weigh in people's reasoning, and enable people to make decisions using their intuition, having gained valuable insights from the reasoning patterns. It remains up to future work to validate the conjecture that, in such complex games, exploration-based advice will be more helpful in practice than direct instruction.

Chapter 7

Conclusion & Extensions

Logic and game theory have been used successfully in the design of computer agents. These theories offer a principled approach for computers to make decisions and communicate with each other. Each, however, faces challenges in environments characterized by large scale and high uncertainty.

Regarding the way people make decisions, psychologists have shown that certain techniques rooted in affect are useful for certain kinds of decisions. This begs the question whether computationally modeling such techniques could provide a similar benefit to computer agents. The preceding chapters of this dissertation answer this question in the affirmative, demonstrating that emotion-inspired techniques can be incorporated into the design of computer agents in a variety of beneficial ways.

First, it shows how to improve agent performance by incorporating analogues of cognitive functions associated with human emotions into an artificial agent's decision-making. It defines computational operators inspired by emotions. These operators are used to re-prioritize an agent's goals in complex uncertain environments, resulting

in improved performance relative to many existing approaches.

Second, it shows how to use an analogue of the communicative functions of human emotions, by building a domain-independent automatic signaling mechanism. This mechanism improves the ability of agents to make inferences about each other's unobservable characteristics, making them both faster and more accurate. Affective signals result in better coordination among agents in collaborative domains, as well as improved payoffs in both collaborative and competitive domains.

As well as serving as signals within populations of computer agents, emotion expressions can also be utilized in agents' interactions with people. The third contribution of this thesis lies in demonstrating that, by displaying emotion, virtual agent faces can influence people's perceptions of a system's trustworthiness. In the context of a computer-human negotiation for allocating resources, people found computer partners that used emotion expressions to be more trustworthy, and in particular when those expressions matched the agents' negotiation strategy. An agent's perceived trustworthiness significantly influenced people's willingness to interact with it in the future.

Finally, this thesis considers strategic environments and makes three more contributions by using the theory of reasoning patterns of Pfeffer & Gal [103]. The reasoning patterns are used to generate advice for people making decisions in complex games. An empirical study demonstrates that this advice can improve people's decision-making performance, by helping them understand the possible effects of various courses of action. The original theory of reasoning patterns is also extended to Bayesian games, with and without a common prior, for which a new graphical

formalism is presented. This formalism can capture agents' reasoning across possibly inconsistent beliefs sets. Finally, the thesis presents a polynomial-time algorithm to detect a game's reasoning patterns. This algorithm is utilized to simplify games for the purpose of computing a Nash equilibrium.

7.1 Reflections on the Use of Emotions

This thesis has shown the usefulness of computational analogues of human emotion for agent decision-making and communication in complex environments. In this section, a few issues concerning the generalizability of this approach are discussed.

First, how is an affective approach to decision-making different from other standard approaches in AI, such as heuristics? The answer to this question can be traced to the fact that the emotions are not tied to specific domains and can therefore generalize across environments. Affective techniques in people and animals have evolved to address the decision-making and communication needs of these living organisms across a great number of situations. We can thus think of emotion as a "meta-heuristic," consisting of generic behavioral responses which, in a given situation, give rise to particular (domain-specific) heuristics. For instance, the emotion of fear is a broad behavioral pattern aimed at recognizing and responding to danger. In a given context, it manifests as a heuristic that detects threats and prompts the organism towards defensive measures, such as escape or hiding.

Second, which situations are more amenable to an affective approach, and which are best addressed by means of standard AI techniques? As the emotions have evolved in nature to deal with complex and uncertain environments, such domains are a

primary candidate. Other situations might also be useful to consider, however. If an agent is designed to operate in multiple different domains, it might be more efficient for its developer to provide it with an approach that generalizes well across domains, instead of incorporating into its design separate solutions for each environment. Also, if an agent is expected to encounter novel situations, the emotions can be used as "meta-heuristics," in the sense described above. In particular, the emotions could generate a set of heuristics that will give the agent satisfactory behavior, until it has explored the domain and can compute a better, more domain-specific solution.

Finally, what are the challenges in designing affect-based agents for decision-making in the real world? To establish good research standards, I would propose the establishment of standards of computational analogues of emotions. To demonstrate that emotions are of value, and to accelerate the adoption of the method, researchers should not design separate computational analogues for the same emotion to suit their domain's needs, but should rely on broadly accepted definitions. As other approaches, such as game theory, have precise definitions, so should the emotions. Moreover, a set of candidate domains must be identified to act as benchmarks, in the same way "robo-soccer" has been used as a testbed for robotics algorithms.

7.2 Future Possibilities

The driving theme of this thesis is that agents should be able to reason efficiently in a variety of settings, and that reasoning can be done in a qualitatively different way from current AI mechanisms and techniques, by using insights from human cognitive and affective processing. The thesis results suggest additional research possibilities,

and they reveal a number of challenges for future consideration. Three particular ways to extend this initial investigation are outlined below.

Enriching Existing Models

This thesis has presented emotion-inspired computational operators to re-prioritize an agent's goals, as well as an affective signaling mechanism for populations of agents, and has demonstrated the usefulness of these techniques. Both of these approaches have built on previous computational accounts of human emotion, but have restricted their attention to a small set of these emotions. People have a very rich emotional repertoire. Ekman [36] has described a set of *basic* emotions, such as joy, sadness, fear or anger, as well as *secondary* emotions like embarrassment, guilt, vengefulness, etc. It is a question for future study whether computational models of these secondary emotions can be utilized in the design of computational operators for agent decision-making, and whether they could carry helpful information as signals among agents, or between agents and people.

Applications to Machine Learning

Machine learning algorithms face many computational challenges similar to those of decision-making. To address these, researchers have exploited the structure of domains, and have suggested hierarchical [16], object-oriented [69] and graphical models [102], among other approaches. Most of these efforts attempt to discover independencies in the structure of the domain, and thus reduce the size and complexity of the learning problem.

Although psychology and neuroscience have not yet offered a clear picture of

effective learning in people,researchers have demonstrated that emotion is a crucial component of the human learning process [24]. People are more likely to retain events of emotional significance, and their mood determines the way they process new information that contradicts their existing beliefs [42].

These findings suggest that there might be significant opportunities for the use of emotion-inspired concepts in Machine Learning. Intrinsic Reinforcement Learning (IRL) is perhaps the most promising candidate in which to deploy emotion-inspired operators. Standard models of Reinforcement Learning (RL) consist of a family of algorithms that utilize a simple exploration principle, by which agents learn which action is best in each state of the world. IRL algorithms improve upon this basic framework by *shaping* the rewards the agent obtains from the environment during its exploration. In particular, IRL algorithms allow agents to receive an additional internal (intrinsic) reward, along with the standard (extrinsic) reward provided by the environment. In many domains this has resulted in improved learning efficiency and the expansion of RL applicability to harder, more complex domains [119].

Computational models of emotion describe how events in the world can trigger emotional responses in people. Such models could inform the design of general-purpose intrinsic reward functions with desirable properties, as also suggested by Sequeira et al. [114]. These intrinsic rewards could perhaps confer some of the benefits that emotion provides in the process of human learning.

Affective Interfaces for Teamwork, Collaboration and Crowdsourcing

In human-human teamwork and collaboration, the expression of emotions has been shown to significantly influence team dynamics [57, 86]. Positive emotion fosters trust and creativity and motivates people to share their ideas more openly. Negative emotion may also beneficial; expressions of sadness implicitly convey a call for support and assistance that can often lead to problems being addressed sooner; even anger has been argued to play a role in improving collaboration between two people, by breaching an unhealthy relationship and helping form a new one [132].

Recent years have witnessed the emergence of Human Computation—and, in particular, of *crowdsourcing* algorithms—as a solution to many task domains in which people can do better than machines. In crowdsourcing, a group of "workers," who may be financially compensated or not, engage in relatively short-term simple tasks, such as image tagging and audio transcription. Recent efforts have extended the crowdsourcing paradigm to longer-term interactions, lasting hours or days, in which people do not work individually but collaborate with each other [17]. In such settings, the efficiency of a crowdsourcing system may depend on how well these workers perform *as a team*. Within such teams, workers may interact in richer ways and relationships may emerge between them, sharing many of the characteristics of traditional human relationships in the workplace. As this thesis and other work has demonstrated, affective cues may be deployed in those settings by the crowdsourcing algorithm itself or in its communication with workers to constructively build and maintain solid relationships among workers and boost productivity, efficiently address conflicts and disruptions, and maintain team cohesion.

Bibliography

[1] D. Antos, C. De Melo, J. Gratch, and B. Grosz. The influence of emotion expression on perceptions of trustworthiness in negotiation. In *Conference on Artificial Intelligence (AAAI)*, 2011.

[2] D. Antos and A. Pfeffer. Representing bayesian games without a common prior. In *Autonomous Agents and Multi-Agent Systems (AAMAS)*, 2010.

[3] D. Antos and A. Pfeffer. Using emotions to enhance decision-making. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2011.

[4] Ioannis Arapakis, Joemon M. Jose, and Philip D. Gray. Affective feedback: an investigation into the role of emotions in the information seeking process. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR 2008, pages 395–402, New York, NY, USA, 2008. ACM.

[5] Christopher Archibald and Yoav Shoham. Modeling billiards games. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems - Volume 1*, AAMAS '09, pages 193–199, Richland, SC, 2009. International Foundation for Autonomous Agents and Multiagent Systems.

[6] Magda Arnold. *Emotion and Personality*. Columbia University Press, 1960.

[7] Lawrence M. Ausubel and Paul Milgrom. The lovely but lonely vickrey auction. Discussion Papers 03-036, Stanford Instititute for Economic Policy Research, August 2004.

[8] A. Axelrod. *The Evolution of Cooperation, Revised Ed.* Perseus Books Group, 2006.

[9] Sulin Ba and Paul A. Pavlou. Evidence of the effect of trust building technology in electronic markets: price premiums and buyer behavior. *MIS Q.*, 26:243–268, September 2002.

[10] B. Barry, I. Fulmer, and N. Goates. Bargaining with feeling: Emotionality in and around negotiation. *Negotiation Theory and Research, New York Psychology Press*, pages 99–127, 2006.

[11] R. F. Baumeister, K. D. Vohs, N. DeWall, and L. Zhang. How emotion shapes behavior: Feedback, anticipation, and reflection, rather than direct causation. *Personality and Social Psychology Review*, 11(2):167–203, 2007.

[12] James Bennett, Stan Lanning, and Netflix. The netflix prize. In *KDD Cup and Workshop in conjunction with KDD*, 2007.

[13] D. Billings. The first international RoShamBo programming competition. *International Computer Games Association Journal*, 23(1):3–8, 2000.

[14] Ben Blum, Christian R. Shelton, and Daphne Koller. A continuation method for nash equilibria in structured games. *J. Artif. Int. Res.*, 25:457–502, April 2006.

[15] C. Boone, C. H. Declerck, and S. Suetens. Subtle social cues, explicit incentives and cooperation in social dilemmas. *Evolution and Human Behavior*, 29:179–188, 2008.

[16] M. Borga. Hierarchical reinforcement learning. In *ICANN-1993, eds S. Gielen and B. Kappen, Amsterdam*, 1993.

[17] Daren C. Brabham. Crowdsourcing as a model for problem solving. *Convergence: The International Journal of Research into New Media Technologies*, 14(1):75–90, 2008.

[18] S.J. Brams. Game theory and emotions. Technical report, C.V. Starr Center for Applied Economics, New York University, 1995.

[19] Doug Bryan, David Lucking-Reiley, Naghi Prasad, and Daniel Reeves. Pennies from ebay: The determinants of price in online auctions. Working Papers 0003, Department of Economics, Vanderbilt University, November 1999.

[20] P. Carnevale and DG. Pruitt. Negotiation and mediation. *Annual Review of Psychology*, (43):531–582, 1992.

[21] T. N. Cason and T. Sharma. Recommended play and correlated equilibria: a case study. *Economic Theory*, 33(1):11–27, 2007.

[22] Xi Chen and Xiaotie Deng. Settling the complexity of two-player nash equilibrium. In *Foundations of Computer Science, 2006. FOCS 2006. 47th Annual IEEE Symposium on*, pages 261 –272, oct. 2006.

[23] In-Koo Cho and David M. Kreps. Signaling games and stable equilibria. *The Quarterly Journal of Economics*, 102(2):pp. 179–222, 1987.

[24] Sven Ake Christianson. *The Handbook of Emotion and Memory: Research and Theory*. Psychology Press, 1992.

[25] L. Cosmides and J. Tooby. *Evolutionary Psychology Handbook*, chapter Neurocognitive Adaptations Designed for Social Exchange, pages 584–627. Wiley NY, 2005.

[26] L. Cosmides and J. Tooby. *Handbook of Emotions*, chapter Evolutionary Psychology and the Emotions. NY: Guilford, 3rd edition, 2008.

[27] C. Crawford and C. Salmon. *Evolutionary Psychology, Public Policy and Personal Decisions*. Lawrence Erlbaum Associates, Publishers, NJ, 2004.

[28] C. Daskalakis, P. W. Goldberg, and C. H. Papadimitriou. The complexity of computing a nash equilibrium. In *ACM Symposium on Theory of Computing*, 2006.

[29] C. De Melo, P. Carnevale, and J. Gratch. The influence of emotions in embodied agents on human decision-making. In *Intelligent Virtual Agents*, 2010.

[30] C. De Melo and J. Gratch. Expression of moral emotions in cooperating agents. In *Intelligent Virtual Agents*, 2009.

[31] Celso de Melo, Peter Carnevale, and Jonathan Gratch. The effect of expression of anger and happiness in computer agents on negotiations with humans. In *Tenth International Conference on Autonomous Agents and Multiagent Systems*, 2011.

[32] Celso de Melo, Liang Zheng, and Jonathan Gratch. Expression of moral emotions in cooperating agents. In *9th International Conference on Intelligent Virtual Agents*, 2009.

[33] M. Dehghani, P. Khooshabeh, L. Huang, L. Oganseyan, and J. Gratch. Cultural frame-switching using accented spoken language by a virtual character. In *Workshop on Culturally Motivated Virtual Characters, IVA*, 2011.

[34] Eddie Dekel, Drew Fudenberg, and David K. Levine. Learning to play bayesian games. *Games and Economic Behavior*, 46(2):282 – 303, 2004.

[35] A. Drolet and M. Morris. Rapport in conflict resolution: Accounting for how face-to-face contact fosters cooperation in mixed-motive conflicts. *Journal of Experimental Social Psychology*, (36):26–50, 2000.

[36] Paul Ekman. Are there basic emotions? *Psychological Review*, 33(3):550–553, 1992.

[37] Michael S. Fanselow and Greg D. Gale. The amygdala, fear, and memory. *Annals of the New York Academy of Sciences*, 985(1):125–134, 2003.

[38] David Ferrucci. Build watson: an overview of deepqa for the jeopardy! challenge. In *Proceedings of the 19th international conference on Parallel architectures and compilation techniques*, PACT '10, pages 1–2, New York, NY, USA, 2010. ACM.

[39] Shai Fine, Yoram Singer, and Naftali Tishby. The hierarchical hidden markov model: Analysis and applications. *Machine Learning*, 32:41–62, 1998. 10.1023/A:1007469218079.

[40] R. H. Frank. *Feelings and emotions: The Amsterdam symposium, Studies in emotion and social interaction*, chapter Introducing Moral Emotions into Models of Rational Choice, pages 422–440. New York, NY, US: Cambridge University Press, 2004.

[41] N. Frijda. *The Emotions. Studies in Emotion and Social Interaction*. New York: Oxford University Press, 1986.

[42] Nico H. Frijda. *A cognitive dissonance theory perspective on the role of emotion in the maintenance and change of beliefs and attitudes. Emotions and belief: How feelings influence thoughts*. Cambridge University Press, 2000.

[43] Aaron Gage, Robin Murphy, Kimon Valavanis, and Matt Long. Affective task allocation for distributed multi-robot teams, 2004.

[44] Alice Xi Gao and Avi Pfeffer. Learning game representations from data using rationality constraints. In *Proceedings of The Conference on Uncertainty in Artificial Intelligence*, UAI 2010, 2010.

[45] Andrew Gilpin and Tuomas Sandholm. A texas hold'em poker player based on automated abstraction and real-time equilibrium computation. In *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, AAMAS '06, pages 1453–1454, New York, NY, USA, 2006. ACM.

[46] H. Gintis. *Game Theory Evolving*. Princeton University Press, 2000.

[47] Herbert Gintis. *Game Theory Evolving*. Princeton University Press, 2nd edition, 2009.

[48] John C. Gittins. Multi-armed bandit allocation indices. *Wiley-Intersc. Series in Systems and Optimization*, 1989.

[49] Piotr J. Gmytrasiewicz and Prashant Doshi. Interactive pomdps: Properties and preliminary results. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 3*, AAMAS '04, pages 1374–1375, Washington, DC, USA, 2004. IEEE Computer Society.

[50] J. Gratch and S. Marsella. Ema: A computational model of appraisal dynamics. In *European Meeting on Cybernetics and Systems Research*, 2006.

[51] Jonathan Gratch, Ning Wang, Jillian Gerten, Edward Fast, and Robin Duffy. Creating rapport with virtual agents. In Catherine Pelachaud, Jean-Claude Martin, Elisabeth Andre, Gerard Chollet, Kostas Karpouzis, and Danielle Pele, editors, *Intelligent Virtual Agents*, volume 4722 of *Lecture Notes in Computer Science*, pages 125–138. Springer Berlin / Heidelberg, 2007.

[52] V. Groom, J. Chen, T. Johnson, F.A. Kara, and C. Nass. Critic, compatriot, or chump? responses to robot blame attribution. In *Human-Robot Interaction (HRI), 2010 5th ACM/IEEE International Conference on Human-robot interaction*, pages 211 –217, march 2010.

[53] B. Grosz, S. Kraus, S. Talman, and B. Stossel. The influence of social dependencies on decision making: Initial investigations with a new game. In *Autonomous Agents and Multi-Agent Systems*, 2004.

[54] Sudipto Guha, Kamesh Munagala, and Peng Shi. Approximation algorithms for restless bandit problems. *J. ACM*, 58:3:1–3:50, 2010.

[55] M. G. Haselton and T. Ketelaar. *Hearts and Minds: Affective influences on social cognition and behavior*, chapter Irrational Emotions or Emotional Wisdom? The Evolutionary Psychology of Emotions and Behavior. NY: Psychology Press, 2005.

[56] R. Hatfield, J.T. Cacioppo, and RL. Rapson. *Emotion and social behavior*, chapter Primitive emotional contagion, pages 151–177. Sage, Newbury Park, CA, 1992.

[57] A. M. Isen, K. A. Daubman, and A. P. Nowicki. Positive affect facilitates creative problem solving. *Journal of Personality and Social Psychology*, 52(6):1122 – 1131, 1987.

[58] Matthew O. Jackson. *Mechanism Theory*, pages 228–277. October 2000.

[59] Manfred Jaeger. Relational bayesian networks. pages 266–273. Morgan Kaufmann, 1997.

[60] N. Jennings, P. Faratin, A. Lomuscio, S. Parsons, M. Wooldridge, and C. Sierra. Automated negotiation: Prospects, methods and challenges. *Group Decision and Negotiation*, 10(2):199–215, 2001.

[61] Albert Xin Jiang, Kevin Leyton-Brown, and Navin A.R. Bhat. Action-graph games. *Games and Economic Behavior*, 71(1):141 – 173, 2011. ¡ce:title¿Special Issue In Honor of John Nash¡/ce:title¿.

[62] Daniel Kahneman. Maps of bounded rationality: Psychology for behavioral economics. *The American Economic Review*, 93(5):pp. 1449–1475, 2003.

[63] E. Kamar and E. Horvitz. Collaboration and shared plans in the open world: Studies of ridesharing. In *IJCAI*, 2009.

[64] Ece Kamar, Ya'akov Gal, and Barbara J. Grosz. Incorporating helpful behavior into collaborative planning. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems - Volume 2*, AAMAS '09, pages 875–882, Richland, SC, 2009. International Foundation for Autonomous Agents and Multiagent Systems.

[65] Michael Kearns, Michael L. Littman, and Satinder Singh. Graphical models for game theory, 2001.

[66] P. Kenny, T. Parsons, J. Gratch, A. Leuski, and A. Rizzo. Virtual patients for clinical therapist skills training. In C. Pelachaud, Jean-Claude Martin, E. Andre, G. Chollet, K. Karpouzis, and D. Pele, editors, *Intelligent Virtual Agents*, volume 4722 of *Lecture Notes in Computer Science*, pages 197–210. Springer Berlin / Heidelberg, 2007.

[67] P. Kenny, T. Parsons, J. Gratch, and A. Rizzo. Evaluation of justina: A virtual patient with ptsd. In H. Prendinger, J. Lester, and M. Ishizuka, editors, *Intelligent Virtual Agents*, volume 5208 of *Lecture Notes in Computer Science*, pages 394–408. Springer Berlin / Heidelberg, 2008.

[68] Patrick Kenny, Arno Hartholt, Jonathan Gratch, William Swartout, David Traum, Stacy Marsella, and Diane Piepol. Building interactive virtual humans for training environment. In *Interservice/Industry Training, Simulation and Education Conference 2007*, 2007.

[69] D. Koller and A. Pfeffer. Object-oriented bayesian networks. In *Proceedings of the Thirteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI), Providence, Rhode Island*, pages 302–313, 1997.

[70] Daphne Koller, Nimrod Megiddo, and Bernhard von Stengel. Efficient computation of equilibria for extensive two-person games. *Games and Economic Behavior*, 14(2):247–259, June 1996.

[71] Daphne Koller and Brian Milch. Multi-agent influence diagrams for representing and solving games. *Games and Economic Behavior*, 45(1):181 – 221, 2003. ¡ce:title¿First World Congress of the Game Theory Society¡/ce:title¿.

[72] P. Kollock. Social dilemmas: The anatomy of cooperation. *Annual Review of Sociology*, 24:183–214, 1998.

[73] R. S. Lazarus. *Emotion and Adaptation*. Oxford University Press, 1991.

[74] Eun Ju Lee, Clifford Nass, and Scott Brave. Can computer-generated speech have gender? an experimental test of gender stereotype. In *CHI '00 extended abstracts on Human factors in computing systems*, CHI EA '00, pages 289–290, New York, NY, USA, 2000. ACM.

[75] K. Leung, R. Bhagat, N. Buchan, M. Erez, and C. Gibson. Culture and international business: recent advances and their implications for future research. *Journal of International Business Studies*, (36):357–378, 2005.

[76] R. Lewicki, B. Barry, and D. Saunders. *Negotiation*. McGraw-Hill, 2010.

[77] R. Lin and S. Kraus. Can automated agents proficiently negotiation with humans? *Communications of the ACM*, 53(1):78–88, 2010.

[78] R. Lin, Y. Oshrat, and S. Kraus. Investigating the benefits of automated negotiations in enhancing people's negotiation skills. In *Autonomous Agents and Multi-Agent Systems (AAMAS)*, 2009.

[79] George F. Luger and William A. Stubblefield. *Artificial Intelligence: Structures and Strategies for Complex Problem Solving*. Addison-Wesley, 1998.

[80] P. Maes, R. Guttman, and A. Moukas. Agents that buy and sell. *Communications of the ACM*, 42(3):81–91, 1999.

[81] Stacy Marsella, Jonathan Gratch, and Paolo Petta. *A blueprint for a affective computing: A sourcebook and manual*, chapter Computational Models of Emotions. Oxford: Oxford University Press, 2010.

[82] R. McKelvey and T. Palfrey. Quantal response equilibria for normal form games. *Games and Economic Behavior*, 10:6–38, 1995.

[83] Brian Milch and Daphne Koller. Ignorable information in multi-agent scenarios, technical report mit-csail-tr-2008-029. Technical report, Massachusetts Institute of Technology, Cambridge, MA, 2008.

[84] Paul Milgrom and John Roberts. Limit pricing and entry under incomplete information: An equilibrium analysis. *Econometrica*, 50(2):443–59, 1982.

[85] Marvin Minsky. *The Emotion Machine: Commonsense Thinking, Artificial Intelligence, and the Future of the Human Mind*. Simon and Schuster, 2007.

[86] M. Morris and D. Keltner. How emotions work: The social functions of emotional expression in negotiations. *Research In Organizational Behavior*, 22:1–50, 2000.

[87] S. Morris. The common prior assumption in economic theory. *In Economics and Philosophy*, 1995.

[88] Roger B. Myerson. *Game Theory: Analysis of Conflict*. Harvard University Press, 1997.

[89] John H. Nachbar. Bayesian learning in repeated games of incomplete information. *Social Choice and Welfare*, 18(2):303–326, 2001.

[90] J. Nash. The bargaining problem. *Econometrica*, (18):155–162, 1950.

[91] C. Nass, I.-M. Jonsson, H. Harris, B. Reaves, J. Endo, and S. Brave. Improving automotive safety by pairing driver emotion and car voice emotion. In *Human Factors in Computing Systems Conference (CHI 2005)*, 2005.

[92] Clifford Nass, Ulla Foehr, and Scott Brave. The effects of emotion of voice in synthesized and recorded speech. In *AAAI Technical Report FS-01-02*, 2005.

[93] Clifford Nass, Ing-Marie Jonsson, Helen Harris, Ben Reaves, Jack Endo, Scott Brave, and Leila Takayama. Improving automotive safety by pairing driver emotion and car voice emotion. In *CHI '05 extended abstracts on Human factors in computing systems*, CHI EA '05, pages 1973–1976, New York, NY, USA, 2005. ACM.

[94] Clifford Nass, Jonathan Steuer, and Ellen R. Tauber. Computers are social actors. In *Conference companion on Human factors in computing systems*, CHI '94, New York, NY, USA, 1994. ACM.

[95] Clifford Nass, Leila Takayama, and Scott Brave. *Advances in Management Information Systems*, chapter Socializing Consistency: From Technical Homogeneity to Human Epitome, pages 373–391. 2006.

[96] R. Nesse. Evolutionary explanations of emotions. *Human Nature*, 1:261–289, 1990.

[97] Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V. Vazirani, editors. *Algorithmic Game Theory*. Cambridge University Press, 2007.

[98] B. O Neill. Technical report, Department of Political Science, Center for International Security and Cooperation, Stanford University, 2000.

[99] K. Oatley and P. N. Johnson-Laird. Towards a cognitive theory of emotions. In *Cognition and Emotion*, volume 1, pages 29–50, 1987.

[100] A. Ortony, G. L. Clore, and A. Collins. *The Cognitive Structure of Emotions*. New York: Cambridge University Press, 1988.

[101] C. H. Papadimitriou and J. N. Tsitsiklis. The complexity of optimal queuing network control. *Math. Oper. Res.*, 24(2):293–305, 1999.

[102] J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. The Morgan Kaufmann series in representation and reasoning. Morgan Kaufmann Publishers, 1988.

[103] A. Pfeffer and Y. Gal. On the reasoning patterns of agents in games. In *Conference on Artificial Intelligence (AAAI)*, 2007.

[104] A. D. Procaccia, J. S. Rosenschein, and A. Zohar. Multi-winner elections: complexity of manipulation, control, and winner-determination. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2007.

[105] Sean R and Eddy. Hidden markov models. *Current Opinion in Structural Biology*, 6(3):361 – 365, 1996.

[106] H. Raiffa. *The Art and Science of Negotiation*. Harvard University Press, 1985.

[107] Leandro C. Rêgo and Joseph Y. Halpern. Generalized solution concepts in games with possibly unaware players. In *Proceedings of the 11th conference on Theoretical aspects of rationality and knowledge*, TARK '07, pages 253–262, New York, NY, USA, 2007. ACM.

[108] Maayan Roth, Reid Simmons, and Manuela Veloso. What to communicate? execution-time decision in multi-agent pomdps. In Maria Gini and Richard Voyles, editors, *Distributed Autonomous Robotic Systems 7*, pages 177–186. Springer Japan, 2006.

[109] Jrn P. W. Scharlemann, Catherine C. Eckel, Alex Kacelnik, and Rick K. Wilson. The value of a smile: Game theory with a human face. *Journal of Economic Psychology*, 22(5):617–640, 2001.

[110] Matthias Scheutz. Agents with or without emotions? In *15th Intl. Florida AI Research Society*, pages 89–93, 2002.

[111] Matthias Scheutz and Paul Schermerhorn. *Handbook of Research on Synthetic Emotions and Sociable Robotics: New Applications in Affective Computing and Artificial Intelligence*, chapter Affective Goal and Task Selection for Social Robots, pages 74–87. IGI Global, 2009.

[112] Matthias Scheutz, Paul Schermerhorn, and James Kramer. The utility of affect expression in natural language interactions in joint human-robot tasks. In *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*, HRI 2006, pages 226–233, New York, NY, USA, 2006. ACM.

[113] John R. Searle. *Speech acts: an essay in the philosophy of language*. Cambridge University Press, 1969.

[114] P. Sequeira, F. Melo, and A. Paiva. Emotion-based intrinsic motivation for learning agents. In *Proceedings of the Conference on Affective Computing and Intelligent Interaction (ACII), Memphis, TN*, 2011.

[115] S. S. Sethuraman. Communicator, programmer, or independent actor: What are computers? In *Communication Division, Conference of the International Communication Association*, 1993.

[116] Ross D. Shachter. Bayes-ball: The rational pastime (for determining irrelevance and requisite information in belief networks and influence diagrams. In *In Uncertainty in Artificial Intelligence*, pages 480–487. Morgan Kaufmann, 1998.

[117] Jiefu Shi and Michael Littman. Abstraction methods for game theoretic poker. In Tony Marsland and Ian Frank, editors, *Computers and Games*, volume 2063 of *Lecture Notes in Computer Science*, pages 333–345. Springer Berlin / Heidelberg, 2001.

[118] Mei Si, Stacy Marsella, and David Pynadath. Thespian: Modeling socially normative behavior in a decision-theoretic framework. In Jonathan Gratch, Michael Young, Ruth Aylett, Daniel Ballin, and Patrick Olivier, editors, *Intelligent Virtual Agents*, volume 4133 of *Lecture Notes in Computer Science*, pages 369–382. Springer Berlin / Heidelberg, 2006.

[119] Satinder Singh, Andrew Barto, and Nuttapong Chentanez. Intrinsically motivated reinforcement learning. In *Proc. of the 18th Annual Conf. on Neural Information Processing Systems (NIPS'04)*, 2005.

[120] R.S. Slivkins and Eli Upfal. Adapting to a changing environment: the brownian restless bandits, 2008.

[121] Robert C. Solomon. *Handbook of Emotions*, chapter The Philosophy of Emotions. NY: Guilford, 3rd edition, 2008.

[122] M.T.J. Spaan, F. A. Oliehoek, and N. Vlassis. Multiagent planning under uncertainty with stochastic communication delays. In *International Conference on Automated Planning*, 2008.

[123] J. Spoor and J. R. Kelly. The evolutionary significance of affect in groups: Communication and group bonding. *Group Processes and Intergroup Relations*, 7(4):398–412, 2004.

[124] Jonathan St.B.T. and Evans. In two minds: dual-process accounts of reasoning. *Trends in Cognitive Sciences*, 7(10):454 – 459, 2003.

[125] K. Sycara and T. Dai. *Handbook of Group Decision and Negotiation*, chapter Agent Reasoning in Negotiation, pages 437–451. Spinger Netherlands, 2010.

[126] Daniel Szer and Francois Charpillet. Improving coordination with communication in multi-agent reinforcement learning. In *Proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence*, ICTAI 2004, pages 436–440, Washington, DC, USA, 2004. IEEE Computer Society.

[127] Alan D. Taylor. The manipulability of voting systems. *The American Mathematical Monthly*, 109(4):pp. 321–337, 2002.

[128] J. Tooby and L. Cosmides. *Handbook of Emotions, Third Edition*, chapter The Evolutionary Psychology of the Emotions and Their Relationship to Internal Regulatory Variables, pages 114–137. Guildford NY, 2008.

[129] A. Tversky and D. Kahneman. *Judgment under uncertainty: heuristics and biases*, pages 32–39. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990.

[130] G. Van Kleef, C. De Dreu, and A. Manstead. The interpersonal effects of anger and happiness in negotiations. *Journal of Personality and Social Psychology*, (86):57–76, 2004.

[131] G. Van Kleef, C. De Dreu, and A. Manstead. An interpersonal approach to emotion in social decision making: The emotions as social information model. *Advances in Experimental Social Psychology*, (42):45–96, 2010.

[132] G. A. Van Kleef and C.K.D. De Dreu. Longer-term consequences of anger expression in negotiation: Retaliation or spillover? *Journal of Experimental Social Psychology*, 46(5):753 – 760, 2010.

[133] Juan D. Velásquez and Pattie Maes. Cathexis: a computational model of emotions. In *Proceedings of the first international conference on Autonomous agents*, AGENTS '97, pages 518–519, New York, NY, USA, 1997. ACM.

[134] Simon A. Williamson, Enrico H. Gerding, and Nicholas R. Jennings. Reward shaping for valuing communications during multi-agent coordination. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems - Volume 1*, AAMAS '09, pages 641–648, Richland, SC, 2009. International Foundation for Autonomous Agents and Multiagent Systems.

[135] F. Wu, S. Zilberstein, and X. Chen. Multi-agent online planning with communication. In *International Conference on Automated Planning*, 2009.

[136] Toshio Yamagishi, Satoshi Kanazawa, Rie Mashima, and Shigeru Terai. Separating trust from cooperation in a dynamic relationship: prisoner's dilemma with variable dependence. *Rationality and society*, 17(3):275–308, 2005.